

Stefan Pasterk

Competency-Based Informatics Education in Primary and Lower Secondary Schools

DISSERTATION

submitted in fulfillment of the requirements for the degree of

Doktor der Naturwissenschaften (Dr. rer. nat.)
Informatikdidaktik

Alpen-Adria-Universität Klagenfurt
Fakultät für Technische Wissenschaften

Supervisor and Evaluator:

Univ.-Prof. Dipl.-Ing. Dr. Andreas Bollin
Alpen-Adria-Universität Klagenfurt
Institut für Informatikdidaktik

Supervisor:

Univ.-Prof. MMag. Dr. Barbara Sabitzer
Johannes Kepler Universität Linz
School of Education - MINT Didaktik

Evaluator:

Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Gerald Futschek
Technische Universität Wien
Information and Software Engineering Group

Klagenfurt, April/2020

AFFIDATIV

I hereby declare in lieu of an oath that

- the submitted academic paper is entirely my own work and that no auxiliary materials have been used other than those indicated,
- I have fully disclosed all assistance received from third parties during the process of writing the thesis, including any significant advice from supervisors,
- any contents taken from the works of third parties or my own works that have been included either literally or in spirit have been appropriately marked and the respective source of the information has been clearly identified with precise bibliographical references (e.g. in footnotes),
- to date, I have not submitted this paper to an examining authority either in Austria or abroad and that
- when passing on copies of the academic thesis (e.g. in bound, printed or digital form), I will ensure that each copy is fully consistent with the submitted digital version.

I understand that the digital version of the academic thesis submitted will be used for the purpose of conducting a plagiarism assessment.

I am aware that a declaration contrary to the facts will have legal consequences.

Stefan Pasterk m.p.

Klagenfurt, April 2020

Dedication

I use this space to express my gratitude to people that were important to me during the completion of the dissertation. At first I want to thank my supervisors Andreas Bollin and Barbara Sabitzer for their trust, their motivating words and their unlimited support. Both of you encouraged me in different ways and to different times and without your support I would not have finished my dissertation. My thanks also to my colleague Max, who always had an open ear for my questions and supported me in several ways. Thank you, for our exciting discussions. Thanks also to Elisa for the support and cooperation during the first years of work on the dissertation. The many projects surrounding the dissertation would not have been possible without the help of many colleagues. Each of them has made his contribution to the completion of the work. So, many thanks to the team of the kindergarten project Alexandra, Katharina, Markus, Nina, and Philipp. It was a great project and the cooperation with you was a pleasure. My thanks to the students Andrii, Vadym, and Olec for their basic work on the GECKO project. A big thank you to Philipp for refining and improving the GECKO system. I love the result. Thanks to Liza and Margo for the development of software tools based on linguistic features. Your support was most welcome. Thanks also to Prof. Günther Fliedl for his advice and feedback concerning natural language processing. The personal support from family and friends is decisive for the successful completion of the dissertation. At first I would like to thank my parents, Ingrid and Josef. Thanks to my father for getting me into computer science and never giving up on asking me

how an exam went. Many thanks to my mother who supported me in everything but considered my work too theoretical. That is why I am dedicating chapter 9 to her. I miss you! Thanks to my two sisters Anna and Michaela for your love, your interest in my work and your support. Many thanks also to my parents-in-law Klaudia and Helmut for your support and the time you spend with our children. I also want to thank my friends Marian and René for listening to my doubts and problems, and giving me advice. The biggest thanks I owe to my beloved wife Sara and my two wonderful daughters Helena and Mona. You three had to give up a lot during my work on the dissertation. Without your love and support I would never have come so far. I appreciate this very much and promise that we will make up for everything. You mean everything to me and I love you very much.

ABSTRACT

Computer science and digital media are playing an increasingly important role in schools and education. In Austria, the subject 'Computer Science' has been anchored in the 9th grade for many years. Additionally, the subject 'Basic Digital Education' is introduced in the first years of secondary education. More and more countries start to teach related topics in the early years of education. For this reason, a number of curricula, educational standards and competency models for computer science and related topics in primary and secondary schools have emerged in recent years. Due to the many different school systems and organizational structures, the documents differ in several aspects. Some of them are described in great detail, while others only give hints about the content to be taught. Also, the basic theory can be different although most models rely on outcome-based education. If teachers, curriculum developers and researchers base their work on more than one specific curriculum, they have their difficulties in the identification of important concepts of computer science related topics. The differences of the educational models make a comparison a complex task as well.

The *graph-based approach* presented in this dissertation provides a possible solution to analyze and compare curricula and to identify central competencies as well as suitable learning paths. For the approach seven selected curricula, educational standards, and competency models are mapped to individual graphs. The nodes represent the learning outcomes, called competencies, and the edges show two different types of dependency relations between them: *expands* and *requires*. Based on graph-theoretic metrics resulting graphs are analyzed and compared. Centrality measures are used to determine *central competencies*. Competencies which occur in similar form in different curricula act as linking points to combine the individual graphs to one generic model of competencies called *Generic, Graph-Based Model for Competencies (GGBMC)*. With this model it is possible to display *learning paths* for targeted competencies and optimize them for different purposes.

In addition to the graph-based approach software tools are presented which automatically support the necessary steps to identify relations between competencies and to combine individual graphs. These tools are based on the linguistic features of the competency formulations and *natural language processing*. To enable the evaluation, organization and application of the approach an online platform called *Graph-based*

*Environment for Competency and Knowledge-Item Organization (GECKO)*¹ is developed and with the project *Lakeside IT-Curriculum* a use case for the graph-based approach is presented. In the first phase a curriculum for kindergarten and the first years of education is developed based on learning paths in the *GGBMC*. The results are discussed and evaluated by educators.

It turns out that the graph-based approach serves successfully for the purpose of analyzing and comparing curricula and their learning outcomes. Additionally, it supports the identification of central competencies and the determination of learning paths for different situations. The developed software tools automatically support the time consuming process of generating and organizing the model and provide an online platform to apply the it.

¹ <https://gecko.aau.at>

ZUSAMMENFASSUNG

Informatik und digitale Medien spielen in Schule und Ausbildung eine immer wichtigere Rolle. In Österreich ist das Fach 'Informatik' seit vielen Jahren in der 9. Schulstufe verankert. Zusätzlich wird das Fach 'Digitale Grundbildung' in den ersten Jahren der Sekundarstufe eingeführt. Immer mehr Länder beginnen mit verwandte Themen in den ersten Jahren der Ausbildung. Aus diesem Grund sind in den letzten Jahren eine Reihe von Lehrplänen, Bildungsstandards und Kompetenzmodellen für Informatik und verwandte Themen in den Primar- und Sekundarschulen entstanden. Aufgrund der vielen unterschiedlichen Schulsysteme und Organisationsstrukturen unterscheiden sich die Dokumente in mehreren Aspekten. Einige davon sind sehr detailliert beschrieben, andere geben nur Hinweise auf die zu vermittelnden Inhalte. Auch die grundlegende Theorie kann unterschiedlich sein, obwohl die meisten Modelle auf ergebnisorientierter Bildung beruhen. Wenn Lehrer, Lehrplanentwickler und Forscher ihre Arbeit auf mehr als einen spezifischen Lehrplan stützen, haben sie ihre Schwierigkeiten bei der Identifizierung wichtiger Konzepte von Themen der Informatik. Die Unterschiede der Bildungsmodelle machen auch einen Vergleich zu einer komplexen Aufgabe.

Der in dieser Dissertation vorgestellte *Graph-basierte Ansatz* bietet eine mögliche Lösung, um Lehrpläne zu analysieren, zu vergleichen und zentrale Kompetenzen sowie geeignete Lernpfade zu identifizieren. Für den Ansatz werden sieben ausgewählte Lehrpläne, Bildungsstandards und Kompetenzmodelle auf einzelne Graphen abgebildet. Die Knoten stellen die Lernziel, Kompetenzen genannt, dar und die Kanten zeigen zwei verschiedene Arten von Abhängigkeitsbeziehungen zwischen ihnen: *expands* und *requires*. Basierend auf graphentheoretischen Metriken werden die resultierenden Graphen analysiert und verglichen. Zur Bestimmung von *Zentralen Kompetenzen* werden Zentralitätsmaße verwendet. Kompetenzen, die in ähnlicher Form in verschiedenen Lehrplänen vorkommen, dienen als Verknüpfungspunkte, um die einzelnen Graphen zu einem generischen Kompetenzmodell namens *Generic, Graph-Based Model for Competencies (GGBMC)* zu kombinieren. Mit diesem Modell ist es möglich, *Lernpfade* für gewünschte Kompetenzen darzustellen und für verschiedene Zwecke zu optimieren.

Zusätzlich zum graphenbasierten Ansatz werden Softwarewerkzeuge vorgestellt, die automatisch die notwendigen Schritte zur Identifizierung von Beziehungen zwischen Kompetenzen und zur Kombination einzelner Graphen unterstützen. Diese

Werkzeuge basieren auf den sprachlichen Merkmalen der Kompetenzformulierungen und des *natural language processing*. Um die Bewertung, Organisation und Anwendung des Ansatzes zu ermöglichen, wird eine Online-Plattform namens *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*² entwickelt. Mit dem Projekt *Lakeside IT-Curriculum* wird ein Anwendungsfall für den graphenbasierten Ansatz vorgestellt. In der ersten Phase wird ein Curriculum für den Kindergarten und die ersten Ausbildungsjahre auf der Basis von Lernpfaden im *GGBMC* entwickelt. Die Ergebnisse werden von den Pädagogen diskutiert und ausgewertet.

Es zeigt sich, dass der graphenbasierte Ansatz erfolgreich zur Analyse und zum Vergleich von Lehrplänen und deren Lernergebnissen dient. Darüber hinaus unterstützt er die Identifizierung von zentralen Kompetenzen und die Bestimmung von Lernpfaden für verschiedene Situationen. Die entwickelten Software-Tools unterstützen automatisch den zeitaufwendigen Prozess der Erstellung und Organisation des Modells und bieten eine Online-Plattform, um es anzuwenden.

² <https://gecko.aau.at>

CONTENTS

1. <i>Introduction</i>	1
1.1 Motivation	1
1.2 Research Questions	2
1.3 Approach and Methods	4
1.4 Structure of the Document	5
 <i>Part I Theory and Background</i>	 9
2. <i>Theoretical Background</i>	11
2.1 Introduction	11
2.2 Definition of Curriculum	11
2.3 Standards-Based Education	12
2.3.1 Educational Standards in English-Speaking Countries	13
2.3.2 Educational Standards in German-Speaking Countries	15
2.3.3 Comparison	17
2.4 Competencies in Education	17
2.4.1 Competency-Based Education in English-Speaking Countries	17
2.4.2 Competency-Based education in German Discussion	20
2.4.3 Comparison	23
2.4.4 Conclusion	24
2.5 A Taxonomy for Standards and Competencies	24
2.5.1 Learning Objectives, Standards and Competencies	24
2.5.2 Bloom's Revised Taxonomy	26
2.6 Computer Science and Digital Literacy	30
2.6.1 Computer Science, Computing, Informatics	30
2.6.2 Digital Literacy and Digital Competence	31
3. <i>Educational Models</i>	35
3.1 Introduction	35
3.2 Research on Computer Science Related Models for Primary Education	35
3.3 Selection of Curricula and Educational Standards	41

3.4	Educational Systems	43
3.5	Selected Educational Models	46
3.5.1	Australia - National Curriculum	46
3.5.2	Austria - digikomp	48
3.5.3	England - National Curriculum	49
3.5.4	Germany - GI Standards	52
3.5.5	Switzerland - National Curriculum	53
3.5.6	United States of America - CSTA Standards	54
3.6	Overview and First Comparison	57
3.7	Conclusion	59
<i>Part II A Graph-Based Approach</i>		61
4.	<i>A Graph-Based Approach</i>	63
4.1	Introduction	63
4.2	Definitions and Theoretical Background	63
4.3	Research on Curriculum Analysis	69
4.4	A Graph-Based Model	73
4.4.1	Attributes for Nodes	73
4.4.2	Types and Attributes for Edges	74
4.4.3	Property Graph	79
4.4.4	Graph Database	84
4.5	Generating the Graphs	85
4.5.1	Identification of Comparable Elements	85
4.5.2	Creation of Relations	87
4.5.3	Reviews and Revision	89
4.6	Generic, Graph-Based Model for Competencies (GGBMC)	90
4.6.1	Steps in the Process	90
4.6.2	Standardization of Competencies	90
4.6.3	Categorization of Competencies	96
4.6.4	Combination to a Generic, Graph-Based Model for Competencies	97
4.7	Conclusion	103
5.	<i>Comparison of Different Educational Models</i>	105
5.1	Introduction	105
5.2	Research in Graph Theoretic Approaches	105
5.3	Graph-Based Metrics	107
5.3.1	Basic Metrics	107
5.3.2	Centrality Measures	108

5.3.3	Structure and Categorization	111
5.4	Results of the Comparison	112
5.4.1	Methods	112
5.4.2	Basic Metric Value Comparison	112
5.4.3	Central Competencies	122
5.4.4	Analysis of Graph-Structure	134
5.5	Comparison of the Categories	141
5.5.1	Computer Science or Digital Literacy	141
5.5.2	Distribution of Categories	147
5.6	Conclusion	150
6.	<i>Identification of Central Competencies and Learning Paths in the GGBMC</i>	155
6.1	Introduction	155
6.2	Analysis of GGBMC	155
6.2.1	Basic Measures	155
6.2.2	Analysis of Intersector Nodes	158
6.3	Identification of Central Learning Outcomes in the GGBMC	161
6.3.1	Centrality Measures	161
6.3.2	Intersector Nodes as Central Competencies	166
6.4	Determination and Optimization of Learning Paths	168
6.4.1	Definitions and Research	168
6.4.2	Methods	170
6.4.3	Exemplary Use Cases and Reflection	171
6.5	Conclusion	176
	<i>Part III Implementation</i>	179
7.	<i>Graph-Based Environment for Competency and Knowledge Item Organiza-</i> <i>tion</i>	181
7.1	Introduction and Existing Research	181
7.2	Tool Description	182
7.2.1	General User Interface	182
7.2.2	Use Cases and Roles	185
7.3	Architecture	190
7.3.1	Technological Overview	190
7.3.2	System Structure	190
7.4	Development	192
7.4.1	Semi-Automated Intersector Node Generation	192
7.4.2	Queries for Centrality Measures	194
7.4.3	Learning Paths Determination	196

7.5	Conclusion	197
8.	<i>(Semi-)Automated Identification of Competencies and Relations</i>	199
8.1	Introduction and Existing Research	199
8.2	Text Analysis-Based Approach	201
8.2.1	Basic Steps	201
8.2.2	Used Libraries	203
8.2.3	Necessary Adaptations	205
8.2.4	Similarity Measures	205
8.3	Results from Text Analysis in Curricula	207
8.3.1	Basic Text-Based Analysis	207
8.3.2	Terms Occurrences in Categories	215
8.4	Synonym Replacement	217
8.4.1	Building a Dictionary	217
8.4.2	Using Dictionaries in Implementation	219
8.5	Generation of Relations	219
8.5.1	Similarity Measure Comparison	219
8.5.2	Improvement of Results	222
8.6	Automated Categorization	227
8.6.1	Introduction	227
8.6.2	Computer Science or Digital Literacy	228
8.6.3	Five Category System	232
8.7	Automated Combination of Competencies	236
8.8	Conclusion	237
9.	<i>Project 'Lakeside IT-Curriculum'</i>	241
9.1	Introduction and Description of the Project	241
9.2	Determination of Competencies and Learning Paths	243
9.2.1	Selection of Competencies	243
9.2.2	Transforming generic Learning Paths to the Project	246
9.2.3	Resulting Curriculum	248
9.3	Opinions from Educators	254
9.4	Comparison to (Semi-)Automated Results	257
9.5	Conclusion	259
<i>Part IV Future Work and Conclusion</i>		261
10.	<i>Future Work</i>	263
10.1	Introduction	263
10.2	Educational Models	263

10.3 The Graph-based Approach	264
10.4 Software Development	264
10.4.1 GECKO	264
10.4.2 Supportive Tools	265
10.5 Projects	267
10.6 Summary	267
11. Conclusion	269
 Appendix	 275
A. Competency Lists	277
A.1 AT: Austria digikomp4 [Dig13b]	277
A.2 AU: Australian National Curriculum [Aus13]	281
A.3 CH: Switzerland Curriculum 21 [Leh14]	288
A.4 DE: GI Standards for Informatics in Primary Education [Ges19]	294
A.5 GB: Model for the National Curriculum England [Ber15]	298
A.6 US1: CSTA K-12 Computer Science Standards 2011 [SCF ⁺ 11]	301
A.7 US2: CSTA K-12 Computer Science Standards 2017 [CST17]	308
B. Intersector Nodes Lists	315
B.1 Data Representation	315
B.2 Digital Applications	316
B.3 Digital Infrastructure	321
B.4 Human Factors and Ethics	324
B.5 Programming and Algorithms	328
C. Kindergarten Questionnaire	333
 Bibliography	 337
List of Figures	352
List of Tables	357
List of Listings	359
List of Definitions	359

1. INTRODUCTION

1.1 Motivation

Information technology and with it computer science and digital media have a deep impact on today's society. Our daily life is affected by these technologies and with that also the education sector. Computer science is playing an increasingly important role in schools and education. This is also reflected by the development and introduction of computer science, computing, informatics, or digital competence (a differentiation of the terms is provided in section 2.6) related curricula in Europe and around the world ([Sch15], [oECEC17]). The countries follow different approaches to incorporate these topics in schools and more and more introduced computer science related topics as mandatory subjects already in primary school. For instance in Switzerland the curriculum for 'Media and Informatics' starts with kindergarten and ends with the 9th school grade. It is competency based and contains competencies to be reached in an own subject as well as competencies which are incorporated into other existing subjects [Leh14]. Due to the many different school systems and organizational structures, a number of curricula, educational standards and competency models for computer science education in primary and secondary schools have emerged in recent years. In Austria, the subject 'Informatics' has been anchored in the 9th school grade (5th year of secondary education) for many years. For experts, just one year offers too little time to accommodate important areas of this field of expertise. For this reason, in Austria the new compulsory subject 'Basic digital education' was introduced in fall 2018 [MPB17]. The exact implementation is planned by schools individually but at least two hours per week for one school year are mandatory requirements and must be fulfilled. The subject may last from 5th till 8th school grade (1st to 4th year of secondary education) and can be implemented as an own subject or be included in existing subjects [BB18]. Covered topics are based on *digikomp*¹, a model for digital competencies [MPB17]. As in many countries, no official curriculum for computer science related topics exist and many approaches, activities and materials are made available through projects and initiatives of universities or third-party providers. For example, the *Computer Science Teachers Association (CSTA)*² in the US, or the *German Informatics Society*

¹ <https://digikomp.at/>

² <https://www.csteachers.org/>

(GI)³ developed several versions of standards for computer science. Other initiatives like the *Bebras contest*⁴ offer a playful, quiz-like approach to many topics related to computer science. Additionally, the *European Computer Driving License (ECDL)*⁵ examines the students' competencies in various application software. This plethora of different specifications and offers makes it difficult for teachers, curriculum developers and researchers to identify important concepts of computer science and to put them together for their respective situations, such as lessons in class. In addition, comparisons and the search for intersections between curricula and projects, due to the obvious differences, are a complex task for which a possible solution based on 'normalization' of competencies and the use of *graphs* is proposed in this dissertation. The main objectives can be summarized with (a) to introduce a technique and framework for comprehensibly evaluating the different curricula, standards and competency models, and (b) to prepare the ground for individually forming a computer science-curriculum in primary and lower secondary schools. Connected to this objectives the resulting research questions are formulated and described in the following section.

1.2 Research Questions

As an important element of curricula, the learning outcomes in form of competencies and standards are in the focus of the research. This work uses the term 'competencies' as general term to refer to the defined learning objectives in selected curricula, educational standards and competency models (described and argued in sections 2.4.4 and 3.7). For this approach it is assumed that not all competencies in an educational model (curriculum, educational standards or competency model) have the same importance or priority. Some are more important to make the model work. These more important competencies are called *central competencies* which are a major part of the following main research question for this dissertation:

RQ0: How can relevant central competencies for computer science education in primary and secondary school be identified and covered?

As the element *central competencies* is very important for the aims of this work, their identification is crucial. For this purpose, and to formally compare different educational models, a graph-based approach is developed. It represents competencies as nodes and their dependencies as the edges of a graph. Based on graph-theoretic methods central nodes can then be identified. Those competencies on which most other competencies depend can for example be considered as central. In addition,

³ <https://gi.de/>

⁴ <https://www.bebbras.org/>

⁵ <http://ecd1.org/>

paths can be calculated which cover these central nodes most efficiently. Which path is the most efficient depends on the requirements that e.g. teachers have in their teaching. This question, like the whole dissertation, is limited to the first years of education, i.e. kindergarten, primary school and the first years of secondary school. The main research question is answered with the help of the following four subquestions.

RQ1: Which relevant competency models, educational standards, curricula and additional third-party projects or initiatives for computer science in primary and secondary education (in the Austrian context) exist?

For the approach of the dissertation information about educational models is essential. Therefore this aspect is covered by the first subquestion. To answer it, relevant literature is collected and investigated. Especially references from related literature are considered to be relevant. Also, discussions and informal interviews with experts and teachers in this field of study are used to get insight information. Regional aspects are also taken into account. The results from this research question are a list and a detailed description of relevant educational models.

RQ2: To what extend is it possible to (semi-)automatically generate graph representations of curricula?

The transfer from the original curricula, educational standards, or competency models to their graph representation needs a lot of time and work. Also, the evaluation of dependency relations between competencies, which is done by experts, consumes a lot of time resources. With different tools based on *graph databases* and *natural language processing* this process can be supported. The question to what extend this is possible is answered by testing these tools and evaluating their results with the help of available data.

RQ3: How and to what extend can curricula specifications be compared based on graph-theory and is it possible to identify central competencies?

The graph representations of the educational models open new possibilities of comparison. Different metrics from graph-theory are used to gather information of the focus, structure, or complexity of the curricula, educational standards, and competency models. In a further step, the very similar competencies are identified to analyze similarities and differences between the different educational models. To answer this question the approach is applied to a set of selected curricula, educational standards, and competency models and the results are described in detail. These results show the new possibilities of this approach and also point out the limits.

RQ4: What is an optimal combination of competencies and competency-clusters to cover central competencies?

Using the results for RQ3, the graph representations of the selected models are combined to one generic graph model for computer science education, called *Generic, Graph-Based Model for Competencies (GGBMC)*. This combination is based on very similar competencies from different curricula, educational standards, and competency models which are used as linking points of the different graph representations. Within this generic model different combinations of competencies are determined and presented. It is used to generate learning paths for target competencies which include all competencies necessary to reach the desired level. The optimal combinations are calculated with the help of different metrics and measures used in RQ3.

1.3 Approach and Methods

This dissertation is based on a combination of different methods, which will lead to a poly-methodological work. As a first step, sources are searched and collected for information on curricula, educational standards, competency models, projects and initiatives related to computer science in primary and secondary education. The basis for the further procedure is laid down by a structured summary and presentation of the data obtained through literature work. Within the individual curricula or projects, for example, relations are established between individual elements, such as between two competencies, in order to use them to create dependency graphs (directed graphs) for existing curricula, educational standards and competency models using the graph database *neo4j*⁶.

The relations between the elements are evaluated by experts for correctness and completeness. In the revision phase, the relations within the curricula, educational standards and competency models are presented to experts. The empirically collected data is validated and the relations in the graphs are corrected accordingly. In order to obtain a general model, elements from the individual, selected curricula, educational standards and competency models are combined to form a graph. This enables a comparison of the interdependencies between different curricula.

Developed tools (*Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*⁷) to support the necessary process in this research are tested and their results evaluated based on the results from manual analysis. Several metrics and methods from information retrieval are used to test the possibilities and limits of the automated approaches.

With the objective of a final evaluation, the graph-based approach is applied to create a curriculum for a local kindergarten project. Based on these experiences, the functionality and benefits of the approach are evaluated and included in the

⁶ <https://neo4j.com/>

⁷ <https://gecko.aau.at>

dissertation as an empirical evaluation. Additionally, the automated determination of learning paths based on graph-theoretic metrics is evaluated using the results from the project.

Initial data are collected through research work in various sources. In order to analyze the selected curricula, they are mapped using a graph database (neo4j). The properties of the individual elements can be collected and evaluated by a precise analysis of the data. The graphic representation form and the graph-theoretical approach open up new possibilities for describing, analyzing and comparing curricula, educational standards, competency models as well as projects and initiatives. In this work, for example, central competencies are to be defined as those competencies whose representative nodes in the graph have an outer degree greater than the upper quartile of the outer degrees of all nodes of these graphs. Accordingly, these are those competencies with the highest number of dependent competencies within a competency model. Other information that can be obtained through this approach would include, for example, the number of nodes without dependencies, sub-graphs on different subject areas, cross-thematic dependencies, the interrelationship of competencies or the occurrence of cycles. An analysis of the resulting data should enable the optimization of the *Generic, Graph-Based Model for Competencies (GGBMC)*. Further data is collected, validated and incorporated into the first versions in the respective revision phases. The expertise will be obtained from secondary school computer science teachers, primary school teachers with an interest in computer science and researchers in the field of computer science didactics.

1.4 Structure of the Document

This dissertation is structured into four parts which form its main body. Each of the parts I, II, III, and IV consists of two to three chapters. Part I has the title '*Theory and Background*' and explores the theory behind the educational models, first definitions, as well as descriptions of relevant curricula, educational standards, and competency models. In part II, the approach to analyze and compare the educational models is presented including results from these processes. Additionally, results for the *Generic, Graph-Based Model for Competencies (GGBMC)* are discussed. Therefore, this part II is called '*A Graph-based Approach*'. Part III, named '*Implementation*', covers documentations of developed tools as well as results from their evaluation. It also includes the description and analysis of the kindergarten project which is based on the graph-based approach. Part IV is called '*Future Work and Conclusion*' and contains possible future application and development for the approach and the supportive tools. It summarizes the conclusions from the individual chapters. In the following paragraphs the parts are explained in more detail including a description of the chapters.

Overview of Part I - Theory and Background

Part I contains the two chapters *2. Theoretical Background* and *3. Educational Models*. As its title indicates, chapter 2 covers the theoretical background of *standards- and competency-based education*. The approaches from different countries are described and compared. Especially, cases from German- and English-speaking countries are part of this discussion. Resulting, relevant elements are defined as they are used in this dissertation. Additionally, with *Bloom's revised taxonomy* a possible way to categorize competencies into complexity-levels is described. A term differentiation in the area of *computer science* and *digital literacy* is also part of chapter 2.

Chapter 3 presents results from related literature research and describes several approaches of comparing computers science related curricula, educational standards, and competency models for early education. It highlights similarities and differences between these approaches and the one developed for this dissertation. Some elements like a categorization system for computer science education are even adopted for the *GGBMC*. Based on the information from literature research, the selection process of educational models is discussed. Crucial factors are described to explain the selection, and the results from this process are presented. For further investigation, a comparison of the educational systems is included followed by a detailed description of the selected curricula, educational standards, and competency models. Finally, some first basic results from the comparison of the models are presented.

Overview of Part II - A Graph-based Approach

Part II consists of the three chapters *4. A Graph-Based Approach*, *5. Comparison of Different Educational Models*, and *6. Identification of Central Competencies and Learning Paths in the GGBMC*. In the first section of chapter 4 the mathematical theory is covered and the required constructs are formally defined. Related work concerning graph representations of curricula is discussed and compared to the graph-based approach of this dissertation. Results from this research lead to a formal description and definition of the graph-based model. The process of generating the graph representations presented in detail showing some examples of the steps taken. Finally, the process to combine the single graph representations to one *Generic, Graph-Based Model for Competencies (GGBMC)* is discussed step by step. For each of the steps several examples are given.

Chapter 5 starts with a discussion of related work for the analysis and comparison of graph-based curricula representations. In the following graph-based metrics are described which add information about e.g. focus, central competencies, structure, or complexity of educational models. First results are presented in form of a comparison of basic values like number of nodes and relations. Identified central competencies are part of the comparison as well, and the results are shown in form of examples. The graph structures are discussed using different criteria like possi-

ble entry points for teaching or not related competencies. The selected educational models are also compared by their foci and the distribution of their competencies over categories.

In chapter 6 the *Generic, Graph-Based Model for Competencies (GGBMC)* is analyzed in detail. One of the basic elements are the *intersector nodes* which play, as linking points, an important role in this analysis and the basic metrics from the previous chapter are used as well. Known centrality measures are applied to the *GGBMC* to identify central competencies. Additionally, the construct of learning paths are discussed and possible applications within the *GGBMC* are presented. The used methods are described and an exemplary use case is shown.

Overview of Part III - Implementation

Part III contains the three chapters 7. *Graph-Based Environment for Competency and Knowledge Item Organization*, 8. *(Semi-)Automated Identification of Competencies and Relations*, and 9. *Project 'Lakeside IT-Curriculum'*. Chapter 7 documents the *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*⁸ a web-based tool to organize graph representations of curricula and to enable expert evaluations and learning path determination. The tool is described by its general user interface, the user roles, and some sample use cases. Screenshots help the reader to get an overview of the interface. Also, the technology behind the system as well as the system's architecture are briefly explained. Some challenging features are discussed and the solutions are presented.

In chapter 8 additional tools to support the process of graph generation are presented and evaluated. The results from a linguistic analysis of the competency formulations lead to approaches to generate relations between competencies, to categorize competencies, and to combine competencies based on natural language processing. Necessary theory for this development is included as well as a documentation about the data preparation called *preprocessing*. The approach to determine dependencies between competencies within the educational models uses similarity measures of the competency formulations. Its results are evaluated using manually added dependency relations. Two different approaches show possibilities to automatically support categorization of the competencies. The results are compared to the opinions of experts which enables an evaluation. Finally, the approach to combine competencies from all selected educational models, which is also based on similarity measures, is discussed.

Chapter 9 describes a use case of the graph-based approach in form of a project to develop a continuous computer science and digital education curriculum starting in kindergarten till the end of secondary education. This curriculum is based on learning paths which are determined by the means of the approach presented in this

⁸ <https://gecko.aau.at>

dissertation. The process of learning paths determination is explained and examples of the results are presented. To evaluate the results, the assessment of the resulting curriculum by the participating educators is presented. These results also enable an evaluation of the automated generation of learning paths based on different metrics.

Overview of Part IV - Future Work and Conclusion

Part IV consists of the chapters *10. Future Work* and *11. Conclusion*. Chapter 10 describes additional possibilities to enhance the presented approach as well as the supportive tools. In the final chapter 11 the conclusions of the chapters are collected and summarized and the research questions are answered.

Part I

THEORY AND BACKGROUND

2. THEORETICAL BACKGROUND

2.1 Introduction

Over the years a lot of different educational approaches came up and new ones replace older ones. In most cases it is a regional dependency, which one is used at which moment. In European regions *competency-based education* is popular at the moment, and curricula are developed based on this theory (e.g. [Leh14]). In the USA *educational standards* have been fostered over years. The term '*educational models*' refers to all these different approaches and is used as an generic term.

This chapter includes important definitions and differentiation of terms and approaches from education theory. As the different educational models differ in several aspects, they also share a lot of similarities, which make a comparison of these models possible. This is shown in the sections 2.3 and 2.4, where *competency-* and *standards-based education* in different regions are described.

Analyzing educational models needs a way of categorization. Bloom's taxonomy is a popular and frequently used theory to classify learning objectives into complexity-levels. In section 2.5 the possible connection between Bloom's taxonomy and the different educational models is described.

In section 2.6 the field of interest is discussed. Over the last years a lot of terms related to computer science subjects in schools became popular. Besides *computer science* or *informatics*, also *digital literacy* or *digital competence* are used. As this work is focusing on computer science related educational models, a clear definition for all of these terms is necessary.

2.2 Definition of Curriculum

Although in countries all over the world curricula are developed to build a general framework for education there exists a broad discussion about the exact meaning of the term and what a curriculum should include [Kel04]. Several definitions of the term 'curriculum' differ in some aspects. Kelly [Kel04, p. 8] sees the curriculum in a very broad sense:

Definition 2.1 (Curriculum 1). *The curriculum is the totality of the experiences the pupil has as a result of the provision made.*

This definition considers different aspects and dimensions of a curriculum and gives a lot of freedom for interpretations. An additional definition comes from Glatthorn et al. [GBWB15, p. 3]:

Definition 2.2 (Curriculum 2). *The curriculum is a set of plans made for guiding learning in the schools, usually represented in retrievable documents of several levels of generality, and the actualization of those plans in the classroom, as experienced by the learners and as recorded by an observer; those experiences take place in a learning environment that also influences what is learned.*

Here a division between the guidance of learning in form of retrievable documents and the actions and experiences in the classrooms are visible. Following Glatthorn et al., those retrievable documents consist of five components [GBWB15, p. 6]:

- a rationale for the curriculum
- the aims, objectives, and content for achieving those objectives
- instructional methods
- learning materials and resources
- tests or assessment methods.

For this dissertation the aims, objectives and the content are of interest, and therefore the term '*curriculum*' will be used to refer to the 'set of plans made for guiding learning' and especially to the formulated learning objectives.

Besides different education systems, also the handling of national curricula differs. Not all countries develop national curricula or if they do, they do not have to be compulsory for federal states. In countries without national curricula for certain subjects, third party associations develop educational models in form of standards or competency models.

2.3 Standards-Based Education

One problem in defining standards, standards-based education or competency and competency-based education is the different use and understanding of these terms in international literature [SW12, KHR08, KMM08]. Generally, educational standards in international context can be seen as learning outcome norms or 'normative requirements used to manage education systems [KAB⁺04]. Great Britain is an exception as their 'standards' represent the actual level of performance reached by the students. In addition to the basic understanding, standards can also differ in various other aspects, e.g. whether they are input- or output-oriented or in their formulation [KAB⁺04]. So, in the next sections, the development of standards-based and competency-based education as relevant for this dissertation, will be discussed.

2.3.1 Educational Standards in English-Speaking Countries

Standards-Based Education in the United States

In the United States' history of education the so called 'standards-based educational reform' was a result of people's and policy's dissatisfaction with the knowledge and skills of their students. Following their opinion, the students in their country would not be able to compete with those from other countries. Higher *standards* for all students should be the solution for this situation [McM08]. The standard-based approach is a form of outcome-based education (OBE) [SW12], which has its origins in the 1950s and was renewed and part of major discussions in the 1980s [MDE13]. OBE was characterized by one of its leading representatives William Spady [Spa94, p. 191] as

Definition 2.3 (Outcome-based education). *a comprehensive approach to organizing and operating an education system that is focused on and defined by the successful demonstrations of learning sought from each student.*

Spady further defines *outcomes* [Spa94, p. 191] as

Definition 2.4 (Outcome). *learning results that are clearly demonstrated at or after the end of an instructional experience. Outcomes can take many forms (from simple to complex) depending on the content, competencies, performance contexts, and consequences embodied in their definition.*

One point of Spady's approach has to be mentioned, as he does not consider *views* or *affective factors* like attitudes, motivation, confidence, and self-concept in his definition of outcomes. He perceives them as essential ingredients to demonstrate successful learning but not as outcomes on their own [Spa94]. This exclusion caused some criticism of Spady's approach [MDE13]. In the same publication the term 'standard' is used and described [Spa94, p. 192] as

Definition 2.5 (Standards 1). *the set of qualities or measures by which performance, skills, or other types of knowledge is judged. These measures can vary along a set of dimensions, including objective-subjective, absolute-relative, substantive-comparative.*

In this definition *qualities* and *measures* play a major role. In form of *competency-based education* OBE became important in higher education especially for medical education [MDE13].

For lower levels of education *standards* were considered to be the answer for the many problems. Horn defines standards [HJ04, p. 1] as

Definition 2.6 (Standards 2). *conceptual or factual criteria representing knowledge, skills, or attitudes that are established by an authority.*

Here, the focus lies on *criteria* coming from the *authority*. As there is no consensus about a clear definition of standards, there is no consensus of what standards should focus on. So, *content standards*, *performance standards*, and *opportunity-to-learn standards* were described and developed [Rav95, KMM08, McM08].

- **Content standards (also curriculum standards):** define essential knowledge, understandings, and skills [McM08]
- **Performance standards:** define degrees of mastery or levels of attainment; describe what kind of performance represents inadequate, acceptable, or outstanding accomplishment [Rav95]
- **Opportunity-to-learn standards:** define the availability of programs, staff, and other resources that schools, districts, and states provide so that students are able to meet challenging content and performance standards [Rav95].

It has to be mentioned, that these three types of standards are related to each other, like e.g. content standards describe what should be taught and performance standards to which level it should be learned [Rav95].

After the controversy concerning standards and standards-based education in the United States, no nationwide standards were developed. Instead all states worked on their own standards for selected subjects and levels including 'binding guidelines for the learning outcomes students are expected to reach' [KMM08].

Educational Standards in Australia

In Australia, education and with it also the development of curricula or educational standards fall into the responsibility of the state. With a document from the Australian's educational ministers in 2008 called 'Melbourne Declaration on Educational Goals for Young Australians' a starting point for standards-based reforms were set [Lam16]. The aim behind this call for reform was to establish teaching standards, curriculum content standards, as well as student performance standards.

As part of the Australian curriculum for each subject and level *achievement standards* are formulated to describe the learning outcomes [Aus13]:

Achievement standards for each learning area or subject describe the learning expected of students at each year level or band of years. Each achievement standard is described in two paragraphs. Typically, the first paragraph describes what students are expected to understand, and the second paragraph describes what students are expected to be able to do having been taught the curriculum content. The set of achievement standards for each learning area or subject describe a broad sequence of expected learning.

2.3.2 Educational Standards in German-Speaking Countries

Learning from the reform in the United States German-speaking countries also started to develop concepts for a standards- as well as competency-based education. Here Germany was in a leading position and Austria and Switzerland based their work on their results [KMM08]. In an expertise from Klieme et al., educational standards are defined as follows [KAB⁺04, p. 15]:

Definition 2.7 (Standards 3). *Educational standards articulate requirements for school-based teaching and learning. They identify goals for pedagogical work, expressed as desired learning outcomes for students. Educational standards thereby translate into concrete terms the educational mission of schools offering a general education.*

Educational standards, as conceived of in this report, draw on general educational goals. They specify the competencies that schools must impart to their students in order to achieve certain key educational goals, and the competencies that children or teenagers are expected to have acquired by a particular grade. These competencies are described in such specific terms that they can be translated into particular tasks and, in principle, assessed by tests.

In the given definition a focus on learning outcomes in form of competencies can be found. The expertise describes three components which are of major interest for the development of these standards:

1. **Social and pedagogical goals:** *educational goals* give general information about the knowledge, abilities, skills, attitudes, values, interests and motivations, and form the basis for educational standards.
2. **Structure of competencies:** educational standards formulate educational goals in form of *competency requirements*, which are organized in *competency models*. These models represent different aspects, levels and processes of development of the competencies and are based on scientific work from didactics and psychology.
3. **Test development concepts and methods:** tasks and assessment programs based on formulated learning outcomes are needed to measure, which competency level students have reached.

Furthermore, seven characteristics of good educational standards are discussed:

1. **Subject-specificity:** Educational standards relate to a specific content area and set out the basic principles of the discipline and/or of the subject in clear terms.

2. **Focus:** The standards do not cover the entire range of the content area or subject in all its ramifications; rather, they concentrate on a core area.
3. **Cumulativity:** Educational standards relate to competencies that have been developed by a certain point in a student's learning biography, and therefore target cumulative, systematically integrated learning.
4. **Binding for all:** They communicate minimum requirements that are expected of all learners. These minimum standards must apply to all students, regardless of school type.
5. **Differentiation:** The standards do not simply set a 'bar', however. Rather, they distinguish between competency levels above and below or prior and subsequent to the achievement of the minimum standard, thus casting light on the learning process and facilitating both the further specification of levels and the differentiation of profiles representative of the additional requirements of a state, a school or a type of school.
6. **Comprehensibility:** The educational standards are formulated in clear, concise and understandable terms.
7. **Feasibility:** The requirements represent a challenge for students and teachers, but can be fulfilled with reasonable effort [KAB⁺04].

As Klieme and Merki compare educational standards from Anglo-American countries with those from German-Speaking countries, they consider educational standards from German-Speaking countries as performance standards including parts of content standards [KMM08]. Performance standards are compared to test scores and content standards to learning goals and content [KAB⁺04]. Although several international standards do not base on competencies and competency models, some of them are developed in a way that they can be interpreted as competency models, as Kliem et al. [KAB⁺04] show.

The German discourse of educational standards started with the bad results in the first PISA phases and resulted in measures to improve the quality of education in 2001 [Her13]. Since 2003, national educational standards for the subjects German and Mathematics have been developed for primary and secondary education. There exist standards for a foreign language for the whole secondary level, where in lower secondary education (middle school) also standards for biology, chemistry and physics are defined. They are defined for students of the ages ten, fifteen, sixteen and nineteen [KMK18].

The development of educational standards in Switzerland was scientifically conducted and followed by empirical tests [KMM08] with the major goal of the 'harmonization of obligatory school' [dkEE18]. It started 2005 with the definition of basic

competencies and competency models. 2011 the first national educational standards for their school language, a foreign language, mathematics, and natural science were presented and describe the basic competencies of the students in the age of eight, twelve and fifteen. These standards influenced the development of different curricula in all cantons in Switzerland [dkEE18].

In Austria, educational standards were tested in form of a pilot study in several selected schools [KMM08] and introduced in 2008. They follow the view of Herzog [Her13] and understand educational standards as 'a certain uniformity in the conditions under which students are taught' [WSBP17]. As the standards focus on the acquirement of basic competency, they support the nation-wide implementation of competency orientation. In connection with the standards, competency models for subjects German (reading and writing) and mathematics were developed for the ages nine to ten and thirteen to fourteen. For the subject English there are standards and a competency model for the thirteen to fourteen years old students [Bun18b, Bun18a].

2.3.3 Comparison

As the standards movement started in the USA and German-speaking approaches learned from their problems, they share a common basis. So for both it is important to define educational goals that should be reached at a given age and that these goals can be measured. Standards in German-speaking countries can be compared to performance standards and also to some extent to content standards, again sharing several elements [KMM08]. On the other hand standards in German-speaking countries are based on competencies and competency models which is not the case in English-speaking approaches. But, in some international cases, like the *Principles and Standards for School Mathematics* of the *National Council of Teachers of Mathematics* published in 2000, the standards can be interpreted as competency models [KAB⁺04]. This fact enables the comparison of existing standards of different countries with each other.

2.4 Competencies in Education

2.4.1 Competency-Based Education in English-Speaking Countries

In English-speaking countries competency-based education has a long history. But nonetheless it is difficult to find definitions of 'competency' or 'competency-based education'. That is because of spelling and interpretation differences. Especially the understanding and usage of the term 'competency' is ambiguous [Dör10] and its definition underlies a lot of discussions, not only because in the English-speaking world a distinction between the terms 'competency' and 'competence' is made.

Competence or Competency

In various dictionaries for the English language the terms '*competency*' and '*competence*' are used synonymously (e.g. Cambridge Dictionary online¹). However, following relevant literature, differences in the geographical use and also in the meaning of these two terms can be found. The term '*competence*' with the plural '*competences*' is primarily used in the English discussion, while in Australia the term '*competency*' with the plural '*competencies*' occur more frequently [Hel07]. A possible distinction of the meaning can be to define *competence*, in a wider or more holistic perspective, as capacity, and *competency*, in a narrower or more atomistic perspective, as disposition needed to handle concrete situations [Car93]. For the field of human performance technology a clear differentiation between *competency model* and *competence model* is essential [Teo06]. Where *competency* refers to characteristics needed for successful performance [Dub98], '*competence equals worthy performance that leads directly to the most efficient accomplishment of organizational goals*' [Teo06, p. 28]. A comparable and very clear distinction comes from Sainz et al. [SGGA⁺10] in the field of knowledge management in electronic systems engineering. They define *competence* [SGGA⁺10, p. 366] as

Definition 2.8 (Competence 1). *the specific level of knowledge, skills, experience, etc. necessary to carry out a defined function*

and *competency* as

Definition 2.9 (Competency 1). *the description of the knowledge, skills, experience, etc. necessary to carry out a defined function.*

In general, it is hard to find a consent about the distinction of the two terms. That is why they are often used as synonyms [Hel07]. In this dissertation the term '*competency*' will be used to refer to the in the next sections defined concept.

Competency-based Education

As mentioned above, in English-speaking countries competency-based education has its roots in outcome-based approaches and is well-established in higher education. The interest for it is rising, considering the movements towards online courses. But again, there neither exists a general valid definition of *competency* nor of *competency-based education* [Bur16]. Le et al. use *competency*-, *mastery*- and *proficiency-based education* interchangeable with *competency education*, but distinguish them from *standards*- and *outcome-based education* [LWS14]. So, Jones et al. provide in their report for the U.S. Department of Education from 2002 following definitions for central concepts [JVP02, p. 7]:

¹ <https://dictionary.cambridge.org/>

- **Skills, Abilities, and Knowledge** are developed through learning experiences, broadly defined to include school, work, participation in community affairs, etc. (although the Competency-Based Initiatives project is focusing on formally organized postsecondary education learning processes).
- **Competencies** are the result of integrative learning experiences in which skills, abilities, and knowledge interact to form bundles that have currency in relation to the task for which they are assembled.

They point out that it is important to distinguish between skills, abilities, knowledge, and competencies which are often used as synonyms. As shown in Fig. 2.1, a hierarchy of these concepts demonstrates their differences.

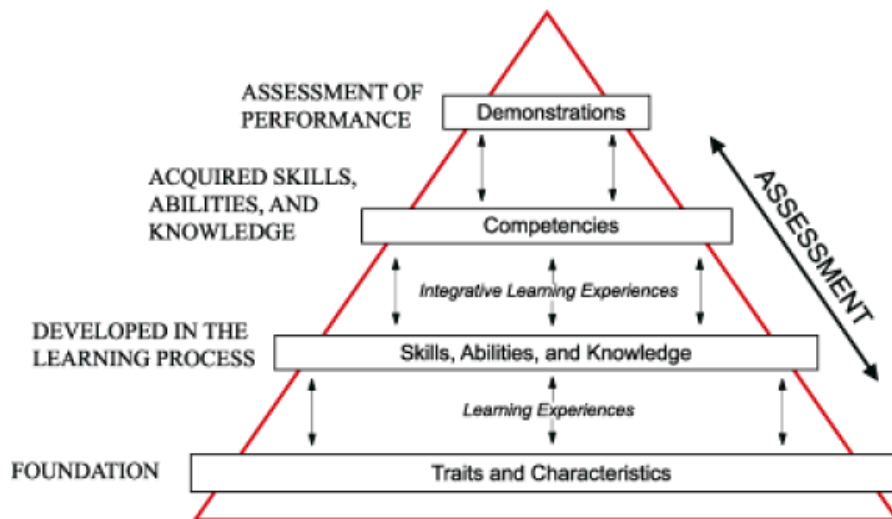


Fig. 2.1: Hierarchical relationship of skills, knowledge and competencies [JVP02, p. 8]

The distinction between *skills*, *knowledge*, and *competencies* is described as '*skills and knowledge are acquired through learning experiences*', but the '*different combinations of skills and knowledge that one has acquired define the competencies that an individual possesses*' [JVP02, p. 8]. Klein-Collins highlights the differences of *learning outcomes* and *competencies*, two terms that are again often used interchangeably [KC12, Bur16]. Following several definitions, learning outcomes '*include specific skills and knowledge, but competencies are at a higher categorical level*' [KC12, p. 9]. Competencies can be applied in more than one specific situation, can be demonstrated and measured, and define different levels of competency. To Klein-Collins competency frameworks '*provide a meaningful description of what a postsecondary degree means in terms of actual student learning*' [KC12].

The main characteristics of competency-based education are summarized by Van der Horst and McDonald [VdHM97] (see also [Bur16]):

- explicit learning outcomes with respect to the required skills and proficiency
- flexibility in time for skill mastery
- a variety of instructional methods to facilitate learning
- criterion-referenced testing to validate the intended outcomes
- certification based on the demonstrated learning outcomes
- adaptable content to 'ensure optimum learner guidance' [Bur16]

2.4.2 Competency-Based education in German Discussion

The definition of essential terms within this field of study is, because of its independent development within several research fields, also a matter of discourses in German speaking regions [Hel07, Dör10]. Because of the high amount of different definitions, this dissertation considers only selected approaches, which are frequently cited in educational contexts. A distinction between the concepts *knowledge*, *skills*, and *competencies* seems necessary, too.

Definition of Competency

In the German discussion concerning educational research fields, one of the most cited definitions for *competency* was published by psychologist Franz Weinert [Wei01b]. The development of national curricula and educational standards in Austria, Germany and Switzerland is based upon Weinert's definition [WSBP17, KAB⁺07, Leh14], which describes *competencies* as

Definition 2.10 (Competency 2). *the cognitive abilities and skills possessed by or able to be learned by individuals that enable them to solve particular problems, as well as the motivational, volitional and social readiness and capacity to use the solutions successfully and responsibly in variable situations.*

The given definition seems very ambiguous as besides cognitive competence it further considers motivational, volitional, and social aspects of competence which are often not covered in educational standards [Kli04]. However, these aspects are not included in recommendations published again by Weinert [Wei99, Wei01a] which focus on the cognitive aspect and adds the needs of a specific context (see also [KMMH07]). Following Weinert, the reduction to the cognitive level limits the concept of competence to '*learned knowledge, skills and corresponding meta-knowledge*'

and is recommended for comparison of school-based performance. This can be compared to the definition of competence by the European Commission [otEU18] in connection with their recommendations concerning key competences for lifelong learning, where '*competences are defined as a combination of knowledge, skills and attitudes*'. Weinert defines *knowledge*, *skills* and *meta-knowledge* as follows [Wei99, p. 35]:

Definition 2.11 (Knowledge 1). ***Knowledge** is the body of understood information possessed by an individual or by a culture.*

Definition 2.12 (Skill 1). ***Skill** is*

1. *an acquired aptitude,*
2. *an ability to perform complex motor and/or cognitive acts with ease, precision, and adaptability to changing conditions.*

Definition 2.13 (Meta-knowledge). ***Meta-knowledge (meta-competencies)** is knowledge about knowledge and deals with the cultural and individual repertoire of rules and regularities for the proper use of the available knowledge:*

1. *declarative meta-knowledge*
2. *procedural meta-knowledge*

The European Commission [otEU18], in connection with their recommendation concerning key competences for lifelong learning, understands knowledge, skills and attitudes as follows [otEU18]:

Definition 2.14 (Knowledge 2). ***Knowledge** is composed of the facts and figures, concepts, ideas and theories which are already established and support the understanding of a certain area or subject.*

Definition 2.15 (Skill 2). ***Skills** are defined as the ability and capacity to carry out processes and use the existing knowledge to achieve results.*

Definition 2.16 (Attitude). ***Attitudes** describe the disposition and mind-sets to act or react to ideas, persons or situations.*

Where *knowledge* and *skills* are similarly defined, *meta-knowledge* and *attitude* represent two different concepts. For Weinert the knowledge about own abilities is one of the three main components of competency. The European Commission considers the behavior of individuals to be important in the context of competency.

Based on Weinert's recommendations Klieme and Leutner [KL06, KAB⁺07] presented their working definition of *competencies* [KHR08, p. 9] as

Definition 2.17 (Competency 3). *context-specific cognitive dispositions that are acquired by learning and needed to successfully cope with certain situations or tasks in specific domains.*

One of the aims behind Weinert's definition is to enable a large-scale assessment of learning outcomes, so measurability should be improved [KHR08, Dör10]. Weinert's definition cited above can be seen as a holistic view of competency, as a general and cross-contextual disposition, although the context aspect is added in other publications. Another perspective would be a cumulative one, where competency is perceived as single and context-related capabilities. In the discussion of suitable measurements for competency this second perspective earned a lot of attention and interest [Hel07].

Subcompetencies and Competency Cluster

A further important definition for this dissertation are *subcompetencies*. Klieme et al. use this term in their expertise without defining it [KAB⁺04]. But in the German version of their expertise they use 'Teilkompetenzen' [KAB⁺07] which means 'subcompetencies', and translate into 'components of competency' [KAB⁺04]. They further connect subcompetencies with different proficiency levels of competency. Winther uses the term 'subcompetencies' (again the German noun 'Teilkompetenzen'), again not giving a definition, to describe how competency areas can be structured [Win10]. In the same context also the term 'competency cluster' (translation from the German word 'Kompetenzbündel') appears, which describes a measurable element of a concrete demanding situation. This measurable element can also be a single competency [Win10]. Following the meaning of 'cluster' and the usage of the term 'competency cluster' it seems to refer to a set of competencies. So subcompetencies and competency cluster describe a similar situation. On the one hand one competency consists of several subcompetencies, and on the other hand a competency cluster consists of several competencies. As some competency formulations in educational documents like e.g. a curriculum or standards seem to include more than one measurable component, in this dissertation these components will be called 'subcompetencies' and the following working definition, combining the two described uses of the term, will be used:

Definition 2.18 (Subcompetency). *Subcompetencies are measurable components of competency and describe different proficiency levels of a competency or a competency area.*

Finding all subcompetencies of a formulated competency needs a detailed analysis of its requirements.

Competency Models

Following the expertise of Klieme et al., *competency models* have the purpose to connect between 'abstract (curricular) educational goals and concrete pools of tasks'. So they provide learning outcomes for a subject that students should reach at a given age or school level as well as the 'routes to knowledge and skills' which should be based on scientific findings [KAB⁺04]. For different purposes different aspects or components of competency are of interest and take a major role in competency assessment and testing. Three major types of competency models were developed [Win10]:

- **Competency structure models (or component models)** in a normative sense describe (cognitive) requirements of a specific subject that learners should have to solve problems and exercises [KAB⁺04, SP06]. On the one hand this should be done by defining competency domains, which represent essential ideas and principles of a discipline, and by structuring these domains into smaller subcompetencies. These subcompetencies can follow taxonomies like Bloom's revised taxonomy from Anderson and Krathwohl [AK01, Win10].
- **Competency level models (or proficiency models)** describe levels of competency with the major goal to enable the measurement of an individual's level of competency [KAB⁺04]. These levels of competency can have various characteristics. For instance it can be distinguished by a cumulative construction or a structural change. In the case of cumulative construction achieved competencies are integrated into higher levels, whereas during structural change once reached competencies are replaced by 'qualitative different competencies' [Abs07].
- **Competency development models** have a focus on the process of the competency development. Based on developmental psychology findings they define structures and development processes of competencies in a specific subject area [Win10].

2.4.3 Comparison

A comparison of the international approaches for competency-based education turns out to be extremely difficult for various reasons. One reason is related to the focus of this dissertation, which lies on lower education, whereas competency-based education in the English-speaking context only appears in research for higher education or vocational training. In the German-speaking discussion a clear distinction is made between competencies for education and for vocational training [KAB⁺04]. Although there are some common elements for the definition of competencies or

competency-based education [Hel07], for lower education different terms are preferred like for instance 'skills' [Dör10]. It seems that in several cases like lower education, a comparison between competency models and educational standards is the only possibility one. That is given by the fact, that there are no competency-based models for computer science in lower education in English-speaking countries.

2.4.4 Conclusion

In most German-speaking regions in the context of lower education Weinert's definition of *competency* [Wei01b] (see definition 2.10) is used. This is a very general view of the concept of competency. Klieme et al. (see definition 2.17) build their definition on that of Weinert, but go a bit more into detail and limit the concept to a specific domain [KHR08]. For the analysis and comparison of educational models in the domain computer science, definition 2.17 by Klieme et al. includes the needed limitations. A further argument is that Klieme et al. consider selected standards as comparable to their understanding of competency models [KAB⁺04]. Therefore, within this work, the definition 2.17 by Klieme et al. is used.

2.5 A Taxonomy for Standards and Competencies

Learning objectives and related taxonomies have a long research tradition. Bloom's taxonomy [BEF⁺56] is a popular example and well known in educational context. Taxonomies are special frameworks helping to arrange elements or categories following a defined continuum [AK01]. That means they support the identification of learning sequences in combination with specified learning objectives. Indeed these learning objectives are related to standards or competencies. So in the next sections at first the concepts of learning objectives, standards and competencies will be compared, and then Bloom's revised taxonomy will be discussed as it also can be applied to standard- or competency-formulations.

2.5.1 Learning Objectives, Standards and Competencies

The goals of the learning process were always an important factor in education and learning objectives should give clear information about them for each subject. One of the most popular and influential definitions of learning objectives goes back to B.J. Bloom and reads as follows:

'explicit formulations of the ways in which students are expected to be changed by the educative process' [BEF⁺56]

This means that learning objectives 'indicate what we want students to learn' or represent 'intended student learning outcomes' [AK01]. Anderson et al. further describe that learning objectives contain two essential elements [AK01]:

- a **verb** to describe the intended cognitive process
- a **noun** to describe the expected knowledge of students

In terms of specificity three different levels of learning objectives can be distinguished:

1. global objectives: broad, complex, and multifaceted learning outcomes, encompassing several more specific learning objectives
2. educational objectives: more specific, focused, and delimited than global objectives, but more general than day-to-day classroom instructions
3. instructional objectives: very specific, day-to-day learning outcomes

A comparison of these levels and their main characteristics can be found in Tab. 2.1.

Following Anderson et al., educational standards also fit into these level of objectives. Depending on aspects like the generality of a standard, if the standard statement includes multiple topics, or if the process is not clearly defined within the standard, standard statements can be categorized into the mentioned levels [AK01].

Tab. 2.1: Characteristics of the different learning objective levels [AK01]

	Level of objective		
	Global	Educational	Instructional
Scope	Broad	Moderate	Narrow
Time needed to learn	One or more years	Weeks or months	Hours or days
Purpose of function	Provide vision	Design curriculum	Prepare lesson plans
Example of use	Plan a multiyear curriculum (e.g. elementary reading)	Plan units of instruction	Plan daily activities, experiences, and exercises

The definitions of standards (see section 2.3) state that they formulate long term objectives. This would place them in the context of global objectives.

Also in the German-speaking discussion, with a strong relation between standards and competencies, their connection to learning objectives is of interest. Learning objectives can therefore be seen on a comparable level with content standards [KAB⁺04] or standards as an extension of learning objectives [Mes06]. In the context of standards Messner describes the specification of learning objectives as follows:

In this new understanding, determining learning objectives means clarifying the competencies in the form of subject-specific situations and minimum requirements and deciding which are to be mastered by the pupils.

This means that in educational standards or competency models, the learning objectives take the form of competency formulations. Following the definitions of learning objectives and competencies it seems that competencies have a more general character, where learning objectives can either be very general or very detailed [Ste10]. Learning objectives can describe a unit in detail, as it is also mentioned in the levels of objectives by Anderson et al. in form of the instructional objectives [AK01]. Consequently, learning objectives can also include very specific knowledge of a subject area.

Studies like the Program for International Student Assessment (PISA) and the Trends on International Mathematics and Science Study (TIMSS) are based on taxonomies which can be compared to Bloom's approach [Hel07].

2.5.2 Bloom's Revised Taxonomy

Since the work of Bloom in the 1940s his taxonomy for educational objectives influenced different developments or scientific studies in the field of education. It differentiates between cognitive, affective, and psychomotor domain.

- **Cognitive domain:** includes objectives which deal with the recall or recognition of knowledge and the development of intellectual abilities and skills.
- **Affective domain:** includes objectives which describe changes in interest, attitudes, and values, and the development of appreciations and adequate adjustment.
- **Psychomotor domain:** includes manipulative or motor-skill area [BEF⁺56].

Bloom's taxonomy focuses on the cognitive domain, which is most relevant for test and curriculum development, and describes six levels of objectives which are arranged in the following order [BEF⁺56, p. 201]:

1. **Knowledge** involves the recall of specifics and universals, the recall of methods and processes, or the recall of a pattern, structure, or setting.
2. **Comprehension** refers to a type of understanding or apprehension, such that the individual knows what is being communicated and can make use of the material or idea being communicated without necessarily relating it to other material or seeing its fullest implications.
3. **Application** represents the use of abstractions in particular and concrete situations.
4. **Analysis** represents the breakdown of a communication into its constituent elements or parts such that the relative hierarchy of ideas is made clear and/or the relations between ideas expressed are made explicit.
5. **Synthesis** represents the putting together of elements and parts to form a whole.
6. **Evaluation** represents judgments about the value of material and methods for given purposes.

This specific ordering is important because each level represents the prerequisite for the next level. In a revised version of the cognitive domain of Bloom's taxonomy by Anderson et al. [AK01] the names of the categories were changed into verbs and the last two were rearranged. So the fifth category changed from *synthesis* to *evaluate* and the sixth category became *create* instead of *evaluation*. Tab. 2.2 shows the six categories of the revised taxonomy including a short description as well as the cognitive processes and alternative names for each of them.

The alternative names give a more detailed view of each of the cognitive processes. Anderson et al. also added a knowledge dimension to Bloom's taxonomy including the four levels *factual knowledge*, *conceptual knowledge*, *procedural knowledge* and *metacognitive knowledge*. The sequence of the four levels represent, as the cognitive levels, that the levels are prerequisite to the next levels [AK01].

- A. **Factual knowledge:** the basic elements students must know to be acquainted with a discipline or solve problems in it
- B. **Conceptual knowledge:** the interrelationships among the basic elements within a larger structure that enable them to function together
- C. **Procedural knowledge:** how to do something, methods of inquiry, and criteria for using skills, algorithms, techniques, and methods

Tab. 2.2: The cognitive process dimension adapted from [AK01]

Category and Cognitive Process	Alternative Name
1. Remember: retrieve relevant knowledge from long-term memory	
1.1 recognizing	identifying
1.2 recalling	retrieving
2. Understand: construct meaning from instructional messages, including oral, written, and graphic communication	
2.1 interpreting	clarifying, paraphrasing, representing, translating
2.2 exemplifying	illustrating, instantiating
2.3 classifying	categorizing, subsuming
2.4 summarizing	abstracting, generalizing
2.5 inferring	concluding, extrapolating, interpolating, predicting
2.6 comparing	contrasting, mapping, matching
2.7 explaining	constructing, models
3. Apply: carry out or use a procedure in a given situation	
3.1 executing	carrying out
3.2 implementing	using
4. Analyze: break material into its constituent parts and determine how the parts relate to one another and to an overall structure or purpose	
4.1 differentiating	discriminating, distinguishing, focusing, selecting
4.2 organizing	finding, coherence, integrating, outlining, parsing, structuring
4.3 attributing	deconstructing
5. Evaluate: Make judgments based on criteria and standards	
5.1 checking	coordinating, detecting, monitoring, testing
5.2 critiquing	judging
6. Create: put elements together to form a coherent or functional whole; reorganize elements into a new pattern or structure	
6.1 differentiating	discriminating, distinguishing, focusing, selecting
6.2 organizing	finding, coherence, integrating, outlining, parsing, structuring
6.3 attributing	deconstructing

D. Metacognitive knowledge: knowledge of cognition in general as well as awareness and knowledge of one's own cognition [AK01]

Together the knowledge and cognitive dimension result in a the well known table shown in Tab. 2.3.

Tab. 2.3: The Taxonomy Table [AK01]

The knowledge dimension	The cognitive process dimension					
	1. Remember	2. Understand	3. Apply	4. Analyze	5. Evaluate	6. Create
A. Factual knowledge						
B. Conceptual knowledge						
C. Procedural knowledge						
D. Meta-cognitive knowledge						

The classification process described by Anerson et al. follows a clear structure. It is based on grammatically correct learning objectives, which have to contain a predicate (verb) and an object (noun). The verb indicates the level within the cognitive dimension and the noun can be used to determine the knowledge dimension. Although Anderson et al. provide a list of action verbs to enable a classification, they clarify that verbs like 'describe' can be classified into different levels depending on the situation. So in some cases it is necessary to know the intention of the writer of a learning objective to correctly classify it [AK01]. A similar situation can occur with nouns which makes an automatic classification impossible [Ste10].

Following Hellwig both understandings of competency, the cumulative and also the holistic view, can be represented in Bloom's taxonomy. From the cumulative perspective competencies of at least partial independent character can be defined on each level. A holistic view requires that all taxonomy levels are accomplished to achieve a competency. This second view seems to be more in Bloom's intention, since he saw the respective level as a prerequisite for the following dimension [Hel07].

2.6 Computer Science and Digital Literacy

In a lot of countries computer science and digital literacy are combined in one subject or curriculum [Sch15]. That is not surprising as both are related to each other, although digital literacy represent practical skills and not a scientific area like computer science [EoIE16]. In this section the terms 'computer science' and 'digital literacy' will be clarified and distinguished.

2.6.1 Computer Science, Computing, Informatics

Defining the term *Computer Science (CS)* needs an additional clarification of the used terminology. Whereas *Computer Science* is a common term in the US, in Europe the term *Informatics* is used broadly and also *Computing Science* can be found [EoIE16]. Formerly, also *Information and Communication Technology (ICT)* was a common term in education, with the varying meaning from 'teaching basic concepts' to 'application of systems' [HAB⁺11]. The UNESCO curriculum from 2002 defines *Informatics (Computing Science)* as follows [Duc02, p. 12]:

Definition 2.19 (Informatics and Computing Science). *The science dealing with the design, realization, evaluation, use, and maintenance of information processing systems, including hardware, software, organizational and human aspects, and the industrial, commercial, governmental and political implications of these.*

Further *Informatics Technology* is defined [Duc02, p. 13]:

Definition 2.20 (Informatics technology). *Informatics technology is defined as the technological applications (artifacts) of informatics in society.*

And finally, *Information and communication technology (ICT)* is described as [Duc02, p. 13]:

Definition 2.21 (Information and communication technology (ICT)). *Information and communication technology, or ICT, is defined as the combination of informatics technology with other, related technologies, specifically communication technology.*

The UNESCO curriculum combines all these three definitions into the term *ICT* which 'will be used, applied and integrated in activities of working and learning on the basis of conceptual understanding and methods of informatics' [Duc02, p. 13]. This is a very general definition and includes a lot of aspects of this research field. In the report of the *Joint Informatics Europe and ACM Europe Working Group* on Informatics Education from 2013 the terms *Informatics* and *Computer Science* are used synonymously and are defined as follows [EoIE16, p. 3]:

Definition 2.22 (Computer Science and Informatics). *Computer Science/Informatics covers the science behind information technology. Informatics is a distinct science, characterized by its own concepts, methods, body of knowledge and open issues. It has emerged, in a role similar to that of mathematics, as a cross-discipline field underlying today's scientific, engineering and economic progress.*

Based on referenced publications [Duc02, HAB⁺11, EoIE16] this work will rely on the definitions from the UNESCO curriculum (see definitions 2.19, 2.20, and 2.21) and use *Computer Science*, *Computing Science*, and *Informatics* synonymously as proposed by the *Joint Informatics Europe and ACM Europe Working Group* [EoIE16].

2.6.2 Digital Literacy and Digital Competence

The impact of information technology on today's society and education makes it necessary to teach topics like correct handling or possible consequences concerning these technologies in schools. Teaching these skills is often a major part of CS classes in school [EoIE16, Sch15] and the terms *Digital Literacy* or also *Digital Competence* became popular to describe them. Although the two terms are often used synonymously they have different meanings [SHLA18].

In 2006 the European Commission published the recommendations on key competences for lifelong learning and defined their fourth key competence *digital competence* as follows:

Digital competence involves the confident and critical use of Information Society Technology (IST) for work, leisure and communication. It is underpinned by basic skills in ICT: the use of computers to retrieve, assess, store, produce, present and exchange information, and to communicate and participate in collaborative networks via the Internet [otEU06].

Martin and Grudziecki follow this definition and regard *digital competence* 'as an underpinning element in digital literacy' [MG06]. They define three levels of *digital literacy* [MG06, p. 255]:

1. Digital competence (skills, concepts, approaches, attitudes, etc.)

2. Digital use (professional/discipline application)
3. Digital transformation (innovation/creativity)

Overall they define *digital literacy* as follows [MG06, p. 255]:

Definition 2.23 (Digital literacy). *Digital Literacy is the awareness, attitude and ability of individuals to appropriately use digital tools and facilities to identify, access, manage, integrate, evaluate, analyse and synthesize digital resources, construct new knowledge, create media expressions, and communicate with others, in the context of specific life situations, in order to enable constructive social action; and to reflect upon this process [MG06].*

Consequently, a strong relationship between the concepts of *digital competence* and *digital literacy* is shown and described as 'digital literacy involves the successful usage of digital competence within life situations' [MG06].

A further definition with high impact was presented in connection with the DIGCOMP framework for Developing and Understanding Digital Competence in Europe [Fer13]:

Digital Competence can be broadly defined as the confident, critical and creative use of ICT to achieve goals related to work, employability, learning, leisure, inclusion and/or participation in society. Digital Competence is a transversal key competence which enables acquiring other key competences (e.g. language, mathematics, learning to learn, cultural awareness). [Fer13]

The DIGCOMP framework [CVP17] includes five competence areas which also give an insight of what *digital competence* represents.

1. Information and data literacy
2. Communication and collaboration
3. Digital content creation
4. Safety
5. Problem solving

Recently also the recommendations on key competences for lifelong learning were revised and aligned with the DIGCOMP and the competence areas [otEU18, p. 4].

Definition 2.24 (Digital competence). *Digital competence involves the confident, critical and responsible use of, and engagement with, digital technologies for learning, at work, and for participation in society. It includes information and data literacy, communication and collaboration, digital content creation (including programming), safety (including digital well-being and competences related to cybersecurity), and problem solving [otEU18].*

This work refers to Martin and Grudziecki’s definition for *digital literacy* (see definition 2.23) [MG06] and to the definition for *digital competence* (see definition 2.24) from the European Commission from 2018 [otEU18].

These definitions show the relations between *digital literacy*, *digital competence* and also *computer science* as the use of information processing systems is included in *computer science* [Duc02]. Therefore, in some cases topic-overlaps make an exact differentiation difficult.

3. EDUCATIONAL MODELS

3.1 Introduction

The masses of different official educational models as well as third-party initiatives and projects complicate a selection of relevant approaches to be included in the study for this dissertation. As different reports show [Sch15, oECEC17], a lot of countries implement computer science related subjects in their curricula and more and more introduce these topics in primary education.

So, in section 3.2 relevant literature for curricula in primary education is summarized, to get information about educational models for computer science. It is followed by section 3.3, that describes the selection process for curricula and educational standards, which is based on previously analyzed related work.

Before the curricula are described in section 3.5, a look is taken at the educational systems of the respective countries in section 3.4.

3.2 Research on Computer Science Related Models for Primary Education

During the last years the research interest for computer science in primary education increased noticeably. That is also reflected by the number of published literature concerning this field of study. The majority of these articles or books focuses on one special newly developed curriculum and explains them in detail. For instance, in the influential article *'Restart: The Resurgence of Computer Science in UK Schools'* Brown et al. describe the situation of the subject *Computing* in the national curriculum in the United Kingdom, discussing different issues and changes to previous approaches [BSCH14]. Berry elaborates in his online resources *'Computing in the national curriculum - A guide for primary teachers'* [Ber13] and *'Quickstart Computing Primary Handbook'* [Ber15] on the curriculum for the subject *Computing* in England. He gives insights to each topic of the curriculum with additional online resources and several hints how they can be taught to primary school children. Similar to the publication of Brown et al. Falkner et al. describe the background and development-process of the Australian curriculum for the subject *Digital Technologies* [FVF14]. In the article *'The Australian Digital Technologies Curriculum:*

Challenge and Opportunity’ they discuss the Australian National Curriculum and the different strands it includes. Further the curriculum is compared to the national curriculum for *computing* in England. The article also contains a literature study to get information about possible concepts and methods for the subject *Digital Technologies* [FVF14]. An overview of the global situation of K-12 education is given by Hubwieser et al. [HGB⁺15]. They do not directly compare the curricula but use articles that discuss the situations in different countries as corpus. There exist only a few analysis and comparisons of different curricula and standard-models for computer science in primary education. One of them is from Barendsen et al. [BMD⁺15]. Their article focuses on computer science concepts in K-9 education, discusses curricula, concepts and assessment from different countries and relates concepts to *Bebras tasks*, an ‘international challenge on Informatics and Computational Thinking’ [oIT]. In their analysis they include the CSTA standards for Computer Science from 2011 [SCF⁺11], the curriculum from the *Computing at School (CAS)* working group from 2012 [Com12], the English national curriculum (EN) from 2014 [Nat14] and Italian guidelines in form of a working document (IT) from 2010. Concerning the analysis the content of these documents is grouped into knowledge categories with the help of open coding [CMM18]. These categories are referred as *knowledge areas* of the *ACM/IEEE Computer Science Curriculum from 2013*, representing ‘guidelines for undergraduate degree programs in Computer Science’ [JTFoCCS13a]. The knowledge areas and the knowledge categories can be seen in Tab. 3.1. The occurrences and distribution of the codes within the curricula documents are calculated and presented to compare the curricula. These results are shown in Tab. 3.2. Barendsen et al. conclude that there is a focus on *algorithmic aspects* in all documents. *Programming* is of similar interest in all four documents, whereas *engineering* does not appear in the Italian documents and only very rarely in national curriculum for England. In the CSTA standards societal aspects are more important than in the documents. *Hardware* and *networks* often appear in the CAS curriculum, whereas the mathematics aspect is, compared to the others, more strongly represented in the Italian guidelines [BMD⁺15]. Additional tasks from the Bebras international competition are analyzed to identify the assessed concepts and the assessment methods [BMD⁺15].

Tab. 3.1: Knowledge categories and referred knowledge areas from the ACM/IEEE Curriculum [BMD⁺15]

Knowledge categories	ACM/IEEE knowledge areas
Algorithms	Algorithms and Complexity (AL) Parallel and Distributed Computing (PD) Algorithms and Design (SDF/AL) Remark: concepts about data structures are covered by data
Architecture	Architecture and Organization (AR) Operating Systems (OS) System Fundamentals (SF)
Modeling	Computational Science (CN) Graphics and Visualisation (GV)
Data	Information Management (IM) Fundamental Data Structures (SDF/IM)
Engineering	Software Engineering (SE) Development Methods (SDF/SE) Remarks: also contains ideas on collaboration; concepts without an engineering component are covered by Programming
Intelligence	Intelligent Systems (IS)
Mathematics	Discrete Structures (DS)
Networking	Networking and Communication (NC)
Programming	Programming Languages (PL) Platform Based Development (PBD) Fundamental Programming Concepts (SDF/PL)
Security	Information Assurance and Security (IAS) Remark: concepts about privacy are covered by Society
Society	Social Issues and Professional Practice (SP)
Usability	Human-Computer Interaction (HCI)

To design a primary school curriculum for computer science and programming, Duncan and Bell [DB15] also compared different curricula. There, the main English-language curricula for the primary school level [DB15], the CSTA K-12 Computer Science Standards [SCF⁺11], the English computing curriculum [Nat14, BSCH14], and the Australian *Digital Technologies* curriculum [Aus13, FVF14] are considered. To identify key ideas, they define the following six categories of themes, based on themes of the chosen curricula and use them as basis for further development [DB15, p. 2]:

- Algorithms:
 - articulating a plan for a program
 - exploring standard problems such as searching or sorting
 - computational complexity (performance)
 - design techniques (such as decomposition and abstraction)
- Programming:
 - implementing a plan/algorithm, debugging, and testing
 - selected programming environments e.g. 'Initial Learning Environments' (drag and drop), general purpose languages, object oriented programming
 - parallel execution
- Data representation:
 - includes binary representation
 - types of data – numbers, text, sound, images etc.
 - encoding the data – compression, error correction, encryption
 - data structures
- Digital devices and infrastructure:
 - includes components, such as CPU, memory, data storage devices
 - input and output devices
 - networks
 - cloud computing
 - troubleshooting devices and configurations
- Digital applications:
 - includes simulation and modeling the real world
 - creating digital content (media) such as documents, images, presentations, web pages, video
 - publishing information
 - collecting and interpreting data (spreadsheets, databases)
 - intelligent systems
- Humans and computers:

- includes digital citizenship (cybersafety, privacy, intellectual property including copyright and licensing, sustainability, ethics, equity, accessibility, legal responsibilities)
- careers (including diversity)
- project management
- usability
- connection to other fields
- keyboard proficiency

Duncan and Bell show, that these categories correspond to those of other educational models including the European Digital Competence framework DIGCOMP [Fer13]. The allocation of the elements of the curricula to the themes shows differences and also similarities of the chosen curricula. The Australian curriculum and the CSTA standards consider the way computers work, like *binary numbers* or *hardware*, in the first years, so with an age of five or six. In England students get in touch with these topics with eleven years. In all three curricula *programming* is a topic from the very beginning. So in the first years students learn about *sequencing* and *turtle graphics*. In the following years *selection*, also called *branching*, and simple *iteration*, also called *repetition*, with counted loops are introduced. Iteration

Tab. 3.2: Relative distribution of the knowledge-category-code occurrences within the curricula [BMD⁺15]

Knowledge categories	CSTA	CAS	EN	IT
Algorithms	16%	23%	30%	30%
Engineering	15%	8%	2%	0%
Architecture	14%	17%	10%	6%
Society	11%	1%	6%	6%
Programming	10%	11%	14%	11%
Intelligence	7%	1%	0%	0%
Modeling	7%	0%	4%	6%
Mathematics	6%	1%	2%	15%
Security	6%	1%	2%	2%
Data	2%	16%	14%	25%
Graphics	2%	0%	0%	0%
Networking	2%	21%	14%	0%
Usability	1%	0%	2%	0%
Rest	0%	0%	0%	0%

with *conditions* and *textbased programming languages* are topics for students aged eleven and older. Topics concerning *safety*, *ethics* or *digital content creation* are covered from the first years and are gradually expanded. *Algorithms* are included in different ways. On the one hand as the 'design of *simple programs*' and on the other as 'algorithms for *standard problems* such as searching and sorting' [DB15]. With an age of eleven in all three curricula students deal with algorithms for standard problems, although the Australian curriculum considers them earlier. A topic that is only covered by the CSTA standards are careers which make use of computing as well as in computing itself [DB15].

In their publication 'Computer science in K-12 school curricula of the 21st century: Why, what and when?', which is based on discussions of the *International Federation of Information Processing (IFIP) Education Community*, Webb et al. analyzed five curricula to get an overview of the meaning of Computer Science in the curricula and propose some principles for curriculum design [WDB⁺17]. A first term clarification is followed by the detailed description of the chosen curricula. In a first step they consider the National curriculum for Computing in England [Nat14, BSCH14], the situation in New Zealand [BAR12, Bel14], the Digital technologies curriculum in Australia [Aus13, FVF14], the situation in Israel [BWD00], and the new informatics curriculum in Poland [SK15]. Summarizing the results, England, Australia and Poland start in early education with elementary school, and teach 'across the content of computer science, IT, digital literacy and computational thinking' [WDB⁺17]. It has to be considered, that the balance between those content areas was not determined during this study. New Zealand and Israel start in high school, and in Israel 'computer literacy' is included for all students, computer science only for some. The following several curricula-reports, which are the CSTA Computer Science standards [SCF⁺11], a report by the Royal Society in UK [Soc] and 'Informatics education: Europe cannot afford to miss the boat' by the Joint Informatics Europe & ACM Europe Working Group [EoIE16], a general consensus exists concerning the key concepts and techniques of computer science. Webb et al. refer to Duncan and Bell [DB15] and their category system. Besides all the content specific competencies the social aspect is included in the Polish curriculum, with project based learning and different roles in groups [WDB⁺17]. Webb et al. published their article 'Tensions in specifying computing curricula for K-12: Towards a principled approach for objectives' in 2018, with an update on the curricula they analyzed [WBD⁺18]. Tab. 3.3 shows the comparison of the now six curricula.

All six countries, Australia, Israel, New Zealand, Poland, UK, and Slovakia, start in an early stage of education with computer science for all students [WBD⁺18].

An overview of the global situation of K-12 education is given by Hubwieser et al. [HGB⁺15] and already summarized by Pasterk et al. [PKB19]. This paragraph is based on the summarization of Pasterk et al.. Hubwieser et al. do not directly compare the curricula but use articles that discuss the situations in different coun-

Tab. 3.3: Comparison of six countries [WBD⁺18]

Theme	Entitlement- who is the COMPUTER SCIENCE curriculum for?	Starting age for COMPUTER SCIENCE
Australia	New curriculum for all	From the first year of school (about 5 years old)
Israel	All must learn Computer Science and technology literacy incorporating computational thinking	Elementary School
NZ	High school subject for seniors from 2011; all children in primary schools from 2018	From 2018 from the first year of school
Poland	High and middle school subject for 20 years. All from 7-11 and an advanced option from 8-12	7 years old
Slovakia	All from elementary school upwards	8 years old
UK	All from elementary school upwards	5 years old

tries as corpus. Following the steps for qualitative text analysis [May15] the corpus is categorized using the tool MaxQDA. In a further step each category is analysed. Especially the categories *Competencies* and *Goals* are of interest. For the elements from the *Competencies* category the verbs are normalized and combined with adverbs and objects. Statements with *and* or *or* are divided into two separate ones. They received 249 competence statements and analyzed knowledge elements like *algorithm* considering used verbs in combinations with them. The statements of the *Goals* category were paraphrased and split if more than one goal was included. Then they were combined and again collected under content topics. Afterwards they compared those new categories and showed which were covered in which countries [HGB⁺15]. Concluding, the authors used a manual qualitative analysis approach to extract, categorize and summarize text passages with different topic foci from research texts.

3.3 Selection of Curricula and Educational Standards

A lot of countries recently developed or implemented curricula for computer science related subjects in early years of education. Following the report 'Computing our

Future' from the European Schoolnet [Sch15] in the year 2015, Estonia, France, Israel, Slovakia, Spain and the United Kingdom already offered computer science related content in primary education. Belgium and Finland planned to integrate it. As several countries like Germany, Switzerland, Sweden or Slovenia did not participate in this report, it does not describe the situation for whole Europe. It either does not include other international countries like the United States or Australia. But it confirms that there exist several educational models for computer science in primary education. Some models come from the government, have an official character and are included in the hour board of the students. Examples for official curricula are the 'Lehrplan 21' from Switzerland with the subject called 'Medien und Informatik (Media and Informatics)' [Leh14] or the Australian Curriculum for 'Digital Technologies' [Aus13]. Others are developed by non-governmental organizations to foster school-autonomous programs or support interested teachers. This is the case for the CSTA (Computer Science Teachers Association) 'Computer Science Standards' from 2011 [SCF⁺11] and 2017 [CST17] as well as the 'Kompetenzen für informatische Bildung im Primarbereich (Competencies for informatics education in primary education)' from the GI (German Informatics Society) [Ges19].

The selection process for curricula and educational standards for this dissertation was preceded by discussions with colleagues in research and education and by reading related literature. With these insights of which impact specific educational models have, a first selection could be made. During the process two aspects were considered:

- Which international educational models for computer science related subjects in primary education exist and have the highest impact in the community?
- Which educational models in German language are of interest?

Both aspects are limited due to language restrictions. Considering the first aspect, as mentioned above, the curricula and educational standards are in most cases developed to be read by teachers in the respective country. That means they are formulated in the country's own language and rarely translated into other languages. This situation implies that the educational models, which can be understood by most people worldwide, have the highest impact. As the English language is one of the most native-spoken languages and known as world's 'lingua franca', the most common second language learned, educational models written in English language can be understood by most people. Following related research in the field of computer science education, the most important English-language curricula are the national curricula from the United Kingdom and from Australia, as well as the 'Computer Science Standards' from the Computer Science Teachers Association (CSTA) of the year 2011 [DB15]. Consequently, these three models were selected and extended by the 2017 revised version of the CSTA 'Computer Science Standards'.

The second aspect considers educational models in German language in particular and displays the interest on local developments. In Germany there is no national curriculum. Each state autonomously develops and implements the curriculum but the German Informatics Society (GI) published their 'Competencies for informatics education in primary education' in 2019 which is selected for this study because of its recency and the focus on computer science topics. In Switzerland the cantons are responsible for education. But since 2014 21 out of 26 cantons agreed to follow the collective curriculum called 'Lehrplan 21'. This curriculum includes the subject 'Media and Informatics', introduced in the first school year and will be part of the study for this dissertation. Austria has no computer science related subject in primary education. But the government offers a competency model for digital competency called 'digikomp' to support interested teachers. This model will be discussed in section 3.5.2 and is one of the selected models to be analyzed in part 2.

The selected educational models will be discussed in more detail in the following sections. But at first the educational systems of the selected countries have to be compared to ensure that they cover the same age groups.

3.4 Educational Systems

An important aspect while comparing and analyzing educational models are the corresponding age groups. In this section the main question is to investigate at which age children go to kindergarten and primary school in the relevant regions.

In **Austria (AT)**, children can start kindergarten in the age of three. The last year before school is compulsory for all children at the age of five. At the age of six, they attend primary school (in German 'Volksschule'), which lasts four years. The first year of primary school represents the first grade, so primary school covers grades 1 to 4. After primary school follows secondary school in form of the *New Secondary School* (in German 'Neue Mittelschule') or the *Academic Secondary School Lower Cycle* (in German 'Allgemein bildende höhere Schule (AHS) Unterstufe'), which again lasts four years and covers grades 5 to 8. At the age of 14, students have to choose between several general and vocational secondary schools. Depending on their choice they finish school with the 12th or 13th grade and an exam (in German 'Matura') at the age of 18 or 19. Compulsory education ends with the ninth grade at the age of 15 [Aus14].

The **Australian (AU)** compulsory education starts at the age of five with the *foundation year* intending to prepare students for their first school year. Depending on the region the foundation year is followed by either six or seven years of primary school, covering grades 1 to 6 or 7. With the age of twelve or thirteen the students enter *lower secondary school*, which lasts three or four years and covers grades 7 or 8 to 10. With the tenth grade, when students are about 16 years old, compulsory

education ends. The optional *senior secondary education* lasts two years and covers grades 11 and 12. It ends with the *senior secondary certificate of education* or *year 12 award* at the age of 18 [Gua17].

In **Switzerland (CH)** kindergarten starts depending on the region at the age of four or five and is compulsory. It lasts one or two years before primary education (in German 'Primarschule') starts at the age of six. Primary school lasts six years and covers grades 1 to 6. With the age of twelve the students enter *lower secondary school* (in German 'Sekundarstufe 1'), which lasts three years. After this period the students are 15 years old and compulsory education ends. At this point students enter upper secondary education (in German 'Sekundarstufe 2') and can choose between general or vocational schools. To help the students in their career choice or to prepare them for general schools, bridging stages (in German 'Brückenangebote') are offered. Schools in upper secondary education last three to four years, cover grades 10 to 12 or 13, and are attended by students until they are 18 or 19 years old [Eur18]. In the new curriculum called 'Lehrplan 21' cycles (in German 'Zyklen') are defined, which cover three to four grades and define a stage of competency development. So kindergarten and grades 1 and 2 are summarized to cycle one, cycle two contains grades 3 to 6, and the first three years of lower secondary education (grades 7 to 9) belong to cycle 3 [Leh14].

In **Germany (DE)**, kindergarten can be attended at an age of three. Compulsory education starts with the age of six and the first year of primary school (in German 'Grundschule'). Primary school lasts four years and covers grades 1 to 4. With an age of ten pupils enter the secondary education, including different school types (e.g. 'Realschule', 'Hauptschule', 'Gymnasium', etc.). The first two years of secondary education are called *orientation stage* (in German 'Orientierungsstufe'). During the orientation stage pupils can switch to another type of school if necessary. After the orientation stage students attend their lower secondary school for three to four years until the age of 15 or 16. It is also possible to start a vocational training with the age of 15. So lower secondary education covers grades 5 to 9 or 10. With grade 11 the upper secondary education starts with the possibilities of general or vocational education. It lasts two to three years and students leave compulsory education with the age of 18 or 19 [Eur18].

In **England (GB)** different options before compulsory education exist and may start before the children are one year old. The compulsory education starts at an age of five with the first year of *primary school*. Primary school last for six years but are divided into *key stage 1*, containing the first two grades, and *key stage 2*, containing grades 3 to 6. At the age of eleven the students enter *secondary schools*, which last five years and are again divided into key stage 3, grades 7 to 9, and key stage 4, grades 10 and 11. After grade 11 the students are 16 years old and the compulsory education ends. The two years of *general secondary* or *vocational secondary education* that follow are only 'part-time' compulsory and can

be combined with courses at working places. School ends with grade 13 at the age of 18 [Eur18].

In the **United States of America (US)** the age to start with compulsory education depends on the state and varies from five to seven years. Only in 15 states kindergarten is compulsory, but in general children start with kindergarten at an age of five. After one year in kindergarten children enter *elementary school*, which can last from five to 8 years, and therefore cover grades 1 to 5 or 8. This also means that students can attend elementary school till they are ten or even thirteen years old. There are different models for the further education, depending on the time spend in elementary school. A five year elementary school is followed by a three year *middle school*, so students are eleven to thirteen years old, and a four year *high school*, students are 14 to 17 years old. After attending a six years elementary school, there exists the possibility to enter a two year *middle school*, followed by a four years *high school*, a three years *junior high school*, followed by a *senior high school*, or a six years *junior and senior high school*. An eight years elementary school is followed by a four years high school. Also the maximum age for compulsory education depends on the state and ranges from 16 to 18 years [Loo18].

Tab. 3.4 shows an overview of the selected education systems. It compares at which ages students start with (min age) and leave (max age) kindergarten, primary and secondary education and the number of grades that are included. Only the compulsory starting ages are considered.

Tab. 3.4: Comparison of relevant Education Systems

	Kindergarten			Primary Education			Secondary Education		
	min age	max age	grades	min age	max age	grades	min age	max age	grades
AT	5	6	1	6	9	4	10	15-19	5-9
AU	5	6	1	6	11-12	6-7	12-13	16-18	3-6
CH	4-5	6	1-2	6	11	6	12	15-19	3-7
DE	/	/	/	6	9	4	10	18-19	8-9
GB	/	/	/	5	10	6	11	16-18	5-7
US	5	6	1	6	10-13	5-8	11-14	17	4-7

As Tab. 3.4 and Fig. 3.1 show, the starting ages differ only slightly as in four of the six regions the children start at an age of five with either kindergarten or primary school. Only in some cantons in Switzerland kindergarten starts at the age of four and in Germany there is no compulsory kindergarten year, so children start at an age of six with primary school. There are more differences between the

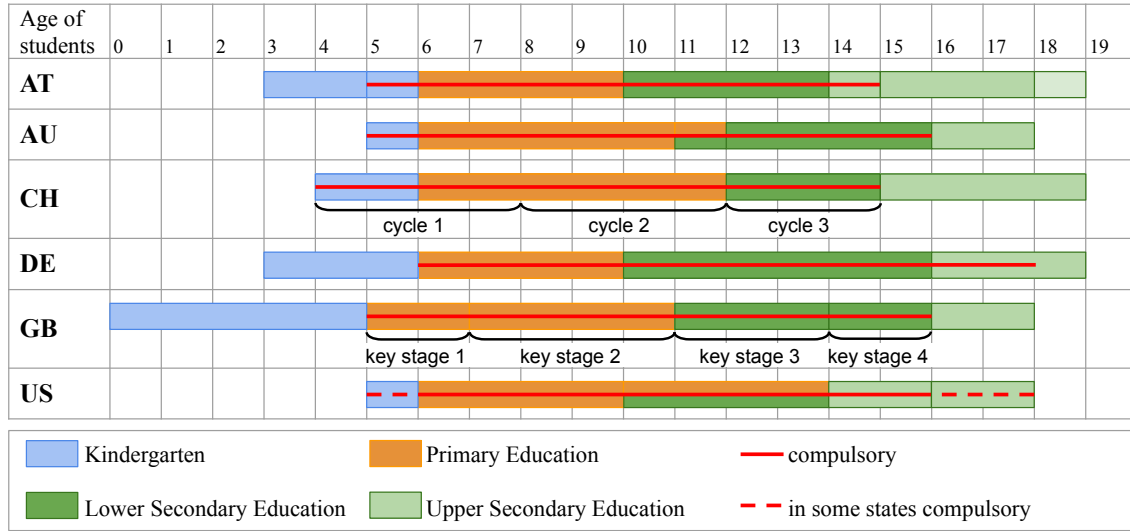


Fig. 3.1: Overview of the educational systems of selected countries

ages when primary education ends. In Austria, Germany and some models of the United States students leave primary schools at an age of ten. Concerning students in England and some regions of Australia, primary school lasts till they are eleven years old, in Switzerland and the other regions of Australia till they are twelve years old. Depending on the chosen model in the United States students can leave primary school with an age of ten, eleven or thirteen.

In the following section the selected curricula, educational standards and competency models will be described, based on already published descriptions by Pasterk and Bollin [PB17b] and Micheuz, Pasterk and Bollin [MPB17].

3.5 Selected Educational Models

3.5.1 Australia - National Curriculum

In Australia, the curriculum for the new learning area *Technologies* was presented in 2013 and represented a combination of the two 'distinct but related' subjects *Design and Technologies* and *Digital Technologies* [FVF14]. Both subjects start with the first school year called Foundation (F) and end at the tenth grade as an elective subject. They are divided into levels that represent two school grades. So the levels F-2, 3-4, and 5-6 cover Australian primary school what means the age range from five to twelve years. In comparison to the Austrian education system the levels 5-6 already cover the first two grades of lower secondary education. Because primary education in Australia lasts till grade 5 or 6, these levels are also considered.

Concerning *Design and Technologies* the main topics are the impact of technolo-

gies on society and related design topics, whereas *Digital Technologies* focuses on the background and the use of information technology. Therefore only the curriculum for *Digital Technologies* will be discussed in this contribution. The curriculum includes a *sequence of content* as well as *achievement standards*, which define what students should be able to do after a given period. The standards contain general learning objectives for the whole learning area *Technologies* and also more specific ones for both subjects *Design and Technologies* and *Digital Technologies* [Aus13]. For analysis only the standards for the learning area *Technologies* and for the subject *Digital Technologies* are considered. Those cover all of the topics from the *sequence of content* and add some more objectives. As the standards are not categorized in the curriculum, they were mapped by the author for comparison. Those, which could not be mapped directly to a content specification, were categorized into the categories of the *sequence of content*.

Overall, the levels F-6 contain 43 achievement standards and 22 content specifications. 13 standards and 6 specifications belong to level F-2, 15 standards and 7 specifications to level 3-4, 15 standards and 9 specifications to level 5-6. The content specifications are divided into the two strands *Knowledge and understanding* and *Processes and production skills*. Both strands are further divided into the following sub-strands, containing the corresponding number of achievement standards and content specifications:

- Digital systems:
 - 3 content specifications
 - 4 achievement standards (4 Digital Technologies)
- Representation of data:
 - 3 content specifications
 - 3 achievement standards (3 Digital Technologies)
- Collecting, managing and analyzing data:
 - 3 content specifications
 - 5 achievement standards (2 Technologies, 3 Digital Technologies)
- Creating digital solutions by investigating and defining:
 - 3 content specifications
 - 7 achievement standards (4 Technologies, 3 Digital Technologies)
- Creating digital solutions by generating and designing:

- 2 content specifications
- 8 achievement standards (6 Technologies, 2 Digital Technologies)
- Creating digital solutions by producing and implementing:
 - 2 content specifications
 - 1 achievement standards (1 Digital Technologies)
- Creating digital solutions by evaluating:
 - 3 content specifications
 - 9 achievement standards (6 Technologies, 3 Digital Technologies)
- Creating digital solutions by collaborating and managing:
 - 3 content specifications
 - 6 achievement standards (2 Technologies, 4 Digital Technologies)

Because in this work mainly outcome-oriented approaches are compared, the *achievement standards* are used in this case. The level F-2 covers the ages five to seven and will correspond to level 1 in this discussion. The years 3-4 and 5-6 represent the age range from eight to twelve and will correspond to level 2. The abbreviation 'AC' will be used to refer to the national curriculum of Australia.

3.5.2 Austria - digikomp

In Austria computer science or digital education is not a subject in primary education on its own. For this reason there is no national curriculum, but there exist suggestions for interested teachers in form of a competency model presented in 2013 by Mulley and Zuliani [MZ13]. It represents the only resource for digital education or computer science related topics for primary education in Austria. The model focuses on digital competency but also considers basic concepts of computer science. Although it does not define 'competency' or 'competency model', it is based on the key competencies for Europe and Digital Competence framework from the EU (Dig-comp) [Dig13b] and it is assumed that also the understanding of competency has been adopted. Overall, it contains 49 competencies, but is not divided into different levels. It covers the four years of primary school with the age range from six to ten years. The following four major topics or strands are included in the model, containing the corresponding number of competencies for each strand:

- Information-technology, humans and society: 16 competencies
- Computer-systems - Usage of digital devices and networks: 13 competencies

- Applications - Digital tools in everyday life: 15 competencies
- Concepts of Computer Science - First steps in informatics: 5 competencies

What distinguishes the digikomp4 model from other approaches is the formulation of the competencies, as they are phrased in the first-person form, how the following example shows:

I am able to cite important application areas of information technology from the living environment. (in German 'Ich kann wichtige Anwendungsgebiete der Informationstechnologie aus der Lebensumwelt anführen.') [Dig13b]

This type of formulation appears in other models like the European digital competence framework DIGCOMP [CVP17], but all other selected approaches use third person definitions.

Digikomp4 as it is called a 'competency model' does not include all aspects of a competency model as it does not define difficulty levels like they are described in 2.4.2. Additionally there is digikomp8 [Dig13c], a digital competency model for lower secondary education, so grades 5 to 8, and digikomp12 [Dig13a], a digital competency model for grades 11 and 12. If digikomp4, digikomp8 and digikomp12 are seen as one competency model the three stages can be interpreted as difficulty levels. So with this bridge a comparison between educational standards is at least possible.

Another problem with the digikomp4 competency model are very concrete and specific formulations of some competencies. Klieme et al. mention that '[...] a competency construct must be sufficiently concrete on the one hand, but should also not be too narrowly defined on the other, as otherwise simple specialist knowledge or isolated skills are unnecessarily labelled as competencies' [KMMH07]. In some cases the digikomp4 model seems too concrete to meet the criteria for competencies, as the following example demonstrates:

I can start and shut down a computer. (in German 'Ich kann einen Computer starten und herunterfahren.') [Dig13b]

Since there is no alternative to this competency model in Austria, it is analyzed and compared in this work despite the open questions. In this dissertation the abbreviation 'DK' will be used to refer to the digikomp4 model.

3.5.3 England - National Curriculum

In 2012 the organization *Computing At School (CAS)* introduced a curriculum [Com12], that highly influenced the 2014 established National Curriculum [Nat14,

BMD⁺15]. The programmes of study for Computing of the National Curriculum in England provides content for the new subject *Computing* replacing *Information and Communication Technology (ICT)*, which was often related to the use of technology instead of learning the concepts of computer science or the creation of software [BSCH14]. Since this new curriculum, in England computer science is taught compulsory from age five to sixteen. The curriculum is divided into four key stages. The first two key stages cover the age range from five to eleven years. Looking at the content defined by the National Curriculum, the following aims can be found:

The national curriculum for computing aims to ensure that all pupils:

- can understand and apply the fundamental principles and concepts of computer science, including abstraction, logic, algorithms and data representation
- can analyse problems in computational terms, and have repeated practical experience of writing computer programs in order to solve such problems
- can evaluate and apply information technology, including new or unfamiliar technologies, analytically to solve problems
- are responsible, competent, confident and creative users of information and communication technology [Nat14].

These are a few very general aims, that are hard to compare with more detailed standards or competencies. Additionally the curriculum defines statements about the content of the subject *Computing*. Overall, the program of study for this curriculum contains 16 statements, six in key stage 1 and ten in key stage 2. What has to be mentioned is, that some of these statements contain more than one knowledge item. This can be shown by the following example statement from key stage 1:

- *Understand what algorithms are; how they are implemented as programs on digital devices; and that programs execute by following precise and unambiguous instructions.* [Ber13]

The whole statement can, for example, be divided into smaller knowledge items by using the given semicolons like

- *Understand what algorithms are.*
- *Understand how algorithms are implemented as programs on digital devices.*
- *Understand that programs execute by following precise and unambiguous instructions.*

If all of the statements would be divided into smaller knowledge items, the curriculum would of course have far more items than the 16 original statements.

The original statements can be categorized into the following three sub-sections, containing the corresponding number of statements for the key stages one and two:

- Computer Science (CS): 8 statements
- Information Technology (IT): 3 statements
- Digital Literacy (DL): 5 statements [Ber13]

These statements within the National Curriculum reflect the subject content, containing the prefix '*Pupils should be taught to ...*' [Nat14]. That means, they are not outcome-based and hardly comparable to standards or competency formulations. In his '*Quickstart Primary Handbook*' Berry includes '*I can ...*'-statements in a '*knowledge and skills audit form*' [Ber15]. These statements also describe smaller knowledge items compared to the original statements in the curriculum, and can be used to record learning progress. For one statement in the curriculum, Berry defines one or more '*I can ...*'-statements. The three following statements have their origin in the above cited curriculum statement.

- *I can explain what an algorithm is.*
- *I can explain how algorithms are implemented as programs on digital devices.*
- *I can explain how programs execute by following precise and unambiguous instructions.*

As these '*I can ...*'-statements are better suited for a comparison with outcome-based approaches, they will be used for further analysis. Because the '*I can ...*'-statements represent smaller knowledge items the numbers of statements changed, as the following list shows.

- Computer Science (CS): 27 statements
- Information Technology (IT): 11 statements
- Digital Literacy (DL): 9 statements [Ber13]

The key stage 1 statements from this curriculum cover the age range from five to seven years and will correspond in further discussion to level 1. Key stage 2 represents statements for students that are eight to eleven years old, and will correspond to level 2. In this dissertation the English national curriculum is abbreviated with 'EC'.

3.5.4 Germany - GI Standards

In Germany the 'Gesellschaft für Informatik (GI)' (the German Informatics Society) already developed and published educational standards for informatics in lower secondary education in 2008 [Ges08] and also for higher secondary education in 2016 [Ges16]. In the spring of 2019 the standards for informatics in primary education called 'Bildungsstandards Informatik für den Primarbereich' [Ges19] were presented. For the first standards for lower secondary education the GI developed a model based on content and process areas which are strongly related to each other [Ges08]. For each of them five areas were defined and can be seen in Fig. 3.2. The competency

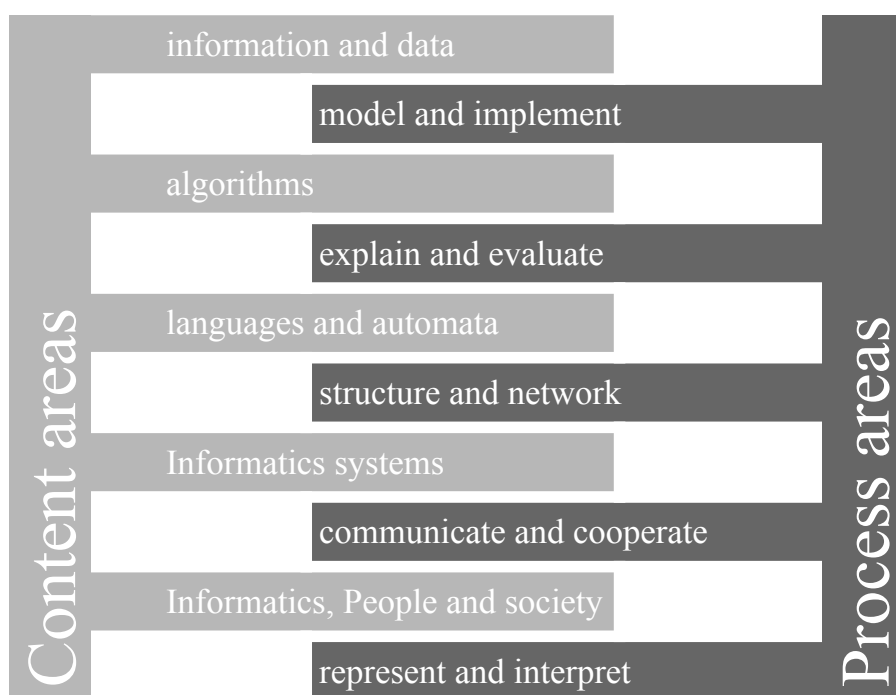


Fig. 3.2: Content and Process areas of the GI Standards [Ges08, Ges19]

expectations (in German 'Kompetenzerwartungen') for primary education are categorized into the content areas but are related to at least one process area [Ges19]. They cover the primary education which in most areas of Germany means the first four years in school that starts with an age of six years and ends with ten years. The standards are divided into two levels, which the students should master after the second year and after the fourth year. Overall, 42 competencies are included, 20 for the first two years and 22 for the third and fourth year. The standards define five content areas with containing a different number of competencies.

- Information and data: 9 competencies

- Algorithms: 7 competencies
- Language and automata: 8 competencies
- Informaticssystems: 9 competencies
- Informatics, humans and society: 9 competencies

These standards are of interest because they have a strong focus on computer science with only few aspects of digital competency. As the only approach, they deal with the topic of automata and modeling of automata, representing an important area of theoretical computer science. Comparable to the Austrian digicomp, the GI standards have to be seen as a recommendation [Ges19]. The competency expectations for primary education are divided into expectations for the second and the fourth grade. Because the age range from grade one and two is from six to eight, the first level of the competencies will correspond to level 1 in further discussion. The competencies of grades three and four cover the ages from eight to ten and will correspond to level 2. For the GI standards the abbreviation 'GI' will be used.

3.5.5 *Switzerland - National Curriculum*

The curriculum for the German speaking cantons in Switzerland is called 'Lehrplan 21' (in English 'curriculum 21'), covers primary and lower secondary education and was presented in 2014. It describes competencies which are based on national educational standards. Because the cantons have the authority over their school systems, they were free to accept it or not. Additionally the cantons were allowed to adapt the original version of this curriculum and therefore 21 of overall 26 cantons accepted the new curriculum. It contains the subject 'Media and informatics (Medien und Informatik)', starting in the first year of primary school. Following the documents of the curriculum [Leh14] the part called 'media' focuses on the understanding and responsible use of media, whereas 'informatics' includes basic concepts of computer science and problem solving. All the other subjects are responsible to foster the application competency needed in their content areas. The curriculum defines three cycles, which can be compared to levels. The first cycle contains kindergarten and the school grades one and two, cycle two grades three to six, and cycle three grades seven to nine. That means the first two cycles cover the primary education which can last in Switzerland four to six years. In the curriculum seven major competencies and content areas are described, four about the 'media' part and three about 'informatics'. For each of this major competencies competency levels, which differ in complexity and focus and represent steps in reaching the major competency, are defined and assigned to the cycle they should be learned in. Overall, 44 competency

levels for cycles one and two exist, 14 for cycle one and 30 for cycle two. The following content areas are part of the curriculum, containing the corresponding number of competency levels for each area:

- Media: Life in media society: 3 competency levels
- Media: Understand media and media products: 7 competency levels
- Media: Produce media and media products: 7 competency levels
- Media: Communicate and cooperate with media: 3 competency levels
- Informatics: Data structures: 7 competency levels
- Informatics: Algorithms: 6 competency levels
- Informatics: Informatics systems: 11 competency levels

Cycle one covers an age range from five to eight years and will therefore correspond to level 1 in this discussion. The second cycle starts with the third year of primary education at the age of nine and ends after four years with the age of twelve and will correspond to level 2. As the curriculum is called 'Lehrplan 21', in this dissertation the abbreviation '21' is used for it.

3.5.6 United States of America - CSTA Standards

In the United States of America computer science in primary education as part of K-12 education was already considered in the first *ACM model curriculum for K-12 computer science* from 2003 [Tuc03, BMD⁺15]. It distinguished between the four levels (1) *Foundations of Computer Science*, (2) *Computers Science in the Modern World*, (3) *Computers Science as Analysis and Design*, and (4) *Topics in Computer Science*. The first level lasted from kindergarten to grade eight, which includes primary education and contains, besides the use of technologies for learning, topics like binary numbers, algorithms, and fundamental logic [Tuc03]. Since 2003, this model curriculum has been revised and new curricula or educational standards for computer science education in primary and secondary schools have been developed. In 2011 the CSTA presented the *CSTA K-12 Computer Science Standards* [SCF⁺11] which are often referenced in relevant literature [BMD⁺15, DB15, HAB⁺11]. The *CSTA K-12 Computer Science Standards* are divided into the three levels (1) *Computer Science and Me*, (2) *Computer Science and Community*, and (3) *Computer Science in the Modern World, Computer Science Concepts and Practices, Topics in Computer Science*. Like the ACM model curriculum they start at the kindergarten and last till the twelfth grade. The standards in the levels K to 6 correspond to the

age range from five to twelve years. Those levels are again divided into Level K-3 and 3-6 standards. Furthermore, the standards are categorized into the following five strands, containing the corresponding number of standards for the levels K to 6:

- Collaboration: 5 standards
- Computational Thinking: 11 standards
- Computing Practice and Programming: 16 standards
- Computers and Communications Devices: 7 standards
- Community, Global, and Ethical Impacts: 6 standards [SCF⁺11]

Overall, the levels K to 6 include 45 standards, 16 for levels K-3 and 29 for levels 3-6. During the analysis of this curriculum two pairs of standards were identified, which are phrased the same or very similar way. The first pair is

Collaboration Grades 3-6:

Use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual and collaborative writing, communication and publishing activities.

Computing Practice and Programming Grades 3-6:

Use technology tools (e.g., multimedia and text authoring, presentation, web tools, digital cameras and scanners) for individual and collaborative writing, communication and publishing activities.

Both standards of the second pair belong to the strand *Computing Practice and Programming* and the grades 3-6, but to different substrands:

Using technology tools for the creation of digital artifacts:

Gather and manipulate data using a variety of digital tools.

Data Collection and Analysis:

Gather and manipulate data using a variety of digital tools.

In this work each of the two pairs are counted as one standard, reducing the overall count of standards to 43, 16 for levels K-3 and 27 for levels 3-6.

The level K-3 represents the age range from five to seven years which will be displayed by level 1 in the further discussion of this contribution. Level 2 will cover the ages from eight to twelve years which corresponds to level 3-6 of the CSTA K-12 Computer Science Standards.

In 2016, these CS standards were revised, based on the in 2016 published K-12 Computer Science Framework [CST16]. These new K-12 Computer Science Standards were published by the CSTA [CST17] in 2017. The framework includes seven core practices including computational thinking, which 'describe the behaviors and ways of thinking that computationally literate students use to fully engage in today's data-rich and interconnected world', and five core concepts, which 'represent major content areas in the field of computer science' [CST16]. The core practices are

1. Fostering and inclusive computing culture
2. Collaborating around computing
3. Recognizing and defining computational problems
4. Developing and using abstractions
5. Creating computational artifacts
6. Testing and refining computational artifacts
7. Communicating about computing

In addition, the core concepts of the framework are also used as categories in the new standards. The CS standards are divided into level 1A with an age range from five to seven, level 1B with an age range from eight to eleven, level 2 with an age range from eleven to fourteen, level 3A with an age range from fourteen to sixteen, and level 1B with an age range from sixteen to eighteen. Considering primary education levels 1A and 1B are of interest for this work. Level 1A contains 18 and level 1B 21 standards. The distribution of the standards among the concepts is as follows.

1. Computing Systems: 6 standards
2. Networks and the Internet: 3 standards
3. Data and Analysis: 5 standards
4. Algorithms and Programming: 18 standards
5. Impacts of Computing: 7 standards [CST16, CST17]

Since the 2011 CS standards had a strong influence on the following approaches, it is included in this work. As also new approaches are of interest, the revised standards from 2017 have been selected, too. To distinguish between both standards the version from 2011 is abbreviated with 'CSTA11' and the version from 2017 with 'CSTA17'.

3.6 Overview and First Comparison

The selected educational models differ in several points as they have to consider the educational system in their countries. So the earliest starting age of students learning computer science related topics begins with four years (Switzerland). The models in Australia, England and USA start at five years, and in Austria and Germany at six years. Most models collect grades to levels (the only exception here is the DK in Austria). Level one lasts in the national curriculum of Australian and Switzerland, as well as in the GI standards in Germany, and the CSTA standards from 2017 till the students are eight years old. In the English national curriculum level one ends with an age of seven, and in the CSTA standards from 2011 with an age of nine. Level two ends in the DK, AC and GI at an age of ten, in the EC and CSTA17 at an age of eleven, and in the 21 and CSTA11 at an age of twelve. An overview of the comparison is presented in Fig. 3.3. Later levels are also included in the figure (green fields), but are not part of this work.

Age of students	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
DK (AT)			dk4				dk8				dk12				
AC (AU)		F-2			3-4		5-6		7-8		9-10				
21 (CH)	cycle 1			cycle 2				cycle 3							
GI (DE)		1-2		3-4		5-7			8-10			11-12			
EC (GB)		key stage 1		key stage 2			key stage 3			key stage 4					
CSTA11 (US)		L1: 3				L1: 6			L2		L3				
CSTA17 (US)		Level 1A			Level 1B			Level 2		Level 3A		Level 3B			

Fig. 3.3: Comparison of covered age-ranges of selected curricula

As the start and end ages differ, also the planned duration of the educational models differ. The DK and GI cover four years, the AC five years, the EC and CSTA17 six years, and the CSTA11 seven years. Finally the 21 lasts eight years, starting with four years. The number of years can be compared in Tab. 3.5. Three of the models (21, Gi, CSTA17) have the same duration for level one as for level two. The AC and CSTA11 consider one level more for level one, and the EC considers two more for level two. In the DK no level separation is done.

Comparing the number of learning objectives again, differences between the models are visible. Five of the selected models (DK, 21, GI, EC, CSTA11) contain between 40 and 50 learning objectives. The CSTA17 is very close to that with 39

Tab. 3.5: Comparison of duration of the selected educational models

	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Duration	4	5	8	4	6	7	6
Duration level 1	/	3	4	2	2	4	3
Duration level 2	4	2	4	2	4	3	3

learning objectives. The shortest one is the AC with 28 learning objectives. In the case of the AC, GI and CSTA17 the learning objectives are more or less equally distributed over both levels. Looking at the 21, the EC and the CSTA11 the numbers of learning objectives from the first level are nearly doubled in the second level. This is shown in Tab. 3.6, where the absolute numbers of learning objectives and their distribution over the two levels is displayed.

Tab. 3.6: Comparison of the absolute numbers of learning objectives

	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
Number of						2011	2017
Learning objectives	49	28	44	42	47	43	39
Level 1 learning objectives	/	13	14	20	17	16	18
Level 2 learning objectives	49	15	30	22	30	27	21

Because of different duration the numbers of learning objectives relative to the covered years make a more detailed comparison possible. The numbers can be found in Tab. 3.7. It can be seen that the two models DK and GI, with the shortest duration, define with the most learning objectives per year, 12.3 and 10.5. The fewest learning objectives per year define the 21, with 5.5, and the AC, with 5.6. Comparing the relative numbers for level one the differences between the highest, with 10 in the GI, and the lowest value, with 3.5 in the 21, is noticeable. This is because level one lasts two years longer in the 21 and level one has a lower number of learning objectives. In level two the differences are not that high. Here, the DK

Tab. 3.7: Comparison of the numbers of learning objectives relative to the covered years

Number of	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Learning objectives per year	12.3	5.6	5.5	10.5	7.8	6.1	6.5
Level 1 learning objectives per year	/	4.3	3.5	10	8.5	4	6
Level 2 learning objectives per year	12.3	7.5	7.5	11	7.5	9	7

and GI show the highest values per year.

3.7 Conclusion

Following the research and related literature in the field of computer science in primary education, seven educational models from six different countries (Australia, Austria, England, Germany, Switzerland, USA) have been selected. The selection process was based on several heuristics, such as impact in related work or language restrictions. After the detailed explanation of the models, there is one competency model (Austria), one national curriculum based on competencies (Switzerland), four educational standards (Australia, Germany, USA), and one national curriculum with additional 'I can ...'-statements (England). As six of the selected models are more or less based on competency-formulations (comparing competencies and standards in section 2.4.4) and 'I can ...'-statements are similar to formulations of competency models (e.g. the Austrian competency model), in this dissertation the generic term *competency* is used to refer to the defined learning objectives of the educational models. After this first comparison of the selected educational models the next part of this dissertation goes more into detail. In the first chapter the graph-based approach will be presented considering related work. After that the second chapter deals with the analysis of the generated graphs and presents results.

Part II

A GRAPH-BASED APPROACH

4. A GRAPH-BASED APPROACH

4.1 Introduction

The approach of this dissertation is based on graph theory, which represents a branch of mathematics. Graphs are well known and often used concepts of mathematics. They are used to 'model systems and the relationships between system parts' [Mar14, p. 8], and are applied in several fields like social network analysis [New07]. As research shows, graphs can also play a role in the representation and analysis of curricula [POM07, Lig10, Ste10, Mar12, Mar14]. In this field of study graphs can be used to visualize relations and dependencies of courses or learning objectives. On behalf of the mathematical origins of graphs, definitions of basic concepts are needed to build a basis for later discussions. Because of the limitation of this work, only for the approach relevant definitions are considered. The definitions and explanations are included in section 4.2. Section 4.3 summarizes related work to the application of graphs in the analysis or comparison of curricula. In section 4.4 the technical basis for building the graphs out of single curricula is explained, including some additional definitions. The process of the graph generation, including the creation of the relations, is described in section 4.5. Finally, in section 4.6, the steps needed to combine the graphs of the selected curricula to one *Generic, Graph-based Model for Competency*, including examples for each step, are presented.

4.2 Definitions and Theoretical Background

There exist a lot of different definitions for graphs and related concepts. Graphs are constructs that connect so called *nodes* or *vertices* with each other using *edges*. *Vertices* are often displayed as points and the *edges* as lines or arrows connecting the points. The most general version of graphs are *undirected* which means the edges have no direction or are bidirectional. Diestel formally defines (*undirected*) *graphs* as follows [Die16, p. 2]:

Definition 4.1 (Graph). A **graph** is a pair $G = (V, E)$ of sets such that $E \subseteq [V]^2$; thus, the elements of E are 2-element subsets of V . To avoid notational ambiguities, we shall always assume tacitly that $V \cap E = \emptyset$. The elements of V are the vertices (or nodes, or points) of the graph G , the elements of E are its edges (or lines).

An example for a graph would be $G = (V, E)$:

$$V = \{x, y, z, u, v, w\}$$

$$E = \{\{x, z\}, \{y, z\}, \{z, u\}, \{u, v\}, \{u, w\}\}$$

In this case the graph has six vertices x, y, z, u, v, w , with x and z , y and z , z and u , u and v , and u and w being connected to each other. A possible graphical representation of this graph is shown in Fig. 4.1.

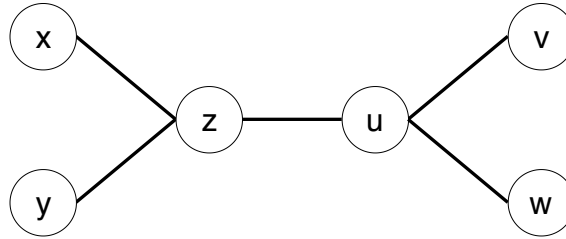


Fig. 4.1: Graphical representation of graph G

With the assumption $V \cap E = \emptyset$ loops, which are defined as follows, are avoided [Mar14, p. 10].

Definition 4.2 (Loop). A **loop** is an edge that connects a vertex to itself.

Loops in graphs representing course or learning objective dependencies are not reasonable, because this would be a dependency on itself. Therefore definition 4.1 is applied for graphs in this dissertation. When graphs are used to model large systems, also a partial view of these systems can be of interest. This is also the case for graphs which represent curricula. Parts of whole graphs are called *subgraphs* and are defined as follows [Die16, p. 3]:

Definition 4.3 (Subgraph). Let $G = (V, E)$ and $G' = (V', E')$ be two graphs. If $V' \subseteq V$ and $E' \subseteq E$, then G' is a **subgraph** of G (and G a **supergraph** of G' , written as $G' \subseteq G$). Less formally, we say that G **contains** G' . If $G' \subseteq G$ and $G' \neq G$, then G' is a **proper subgraph** of G . If $G' \subseteq G$ and G' contains all the edges $xy \in E$ with $x, y \in V'$, then G' is an **induced subgraph** of G ; we say that V' induces or spans G' in G , and write $G' =: G[V']$. Thus if $U \subseteq V$ is any set of vertices, then $G[U]$ denotes the graph on U whose edges are precisely the edges of G with both ends in U .

So a *subgraph* contains only *vertices* and *edges*, which are also part of the original graph. A *proper subgraph* does not contain all *vertices* or not all *edges* of the original graph.

The number of edges of a vertex represents a measure that is important for this work and is called the *degree* [Die16, p. 5].

Definition 4.4 (Degree). The **degree** $d_G(v) = d(v)$ of a vertex v is the number $|E(v)|$ of edges at v ; by our definition of a graph, this is equal to the number of neighbors of v . A vertex of degree 0 is **isolated**. The number $\delta(G) := \min\{d(v) | v \in V\}$ is the **minimum degree** of G , the number $\Delta(G) := \max\{d(v) | v \in V\}$ its **maximum degree**.

With the goal of representing dependencies of learning objectives, the edges of corresponding graphs need to have a direction. Otherwise the information, which learning objective is dependent from the other one, would be lost. Thus, directed edges are required. There exist a lot of types of graphs for different purposes and some of them can be described as follows [RN10]:

- **half-edge graph**: a unary edge (i.e. an edge that 'connects' one vertex) has limited practical application and is primarily discussed in mathematics.
- **directed graph**: orders the vertices of an edge to denote edge orientation.
- **pseudo-graph**: used to denote a reflexive relationship.
- **multi-graph**: allows multiple edges between the same two vertices.
- **simple graph**: the prototypical graph, where an edge connects two vertices and no loops are allowed.
- **weighted graph**: used to represent strength of ties or transition probabilities.
- **vertex-labeled graph**: most graphs make use of labeled vertices (e.g. an identifier).
- **vertex-attributed graph**: used in applications where it is desirable to append non-relational meta-data to a vertex.
- **edge-labeled graph**: used to denote the way in which two vertices are related (e.g. friendships, employment, etc.).
- **edge-attributed graph**: used in applications where it is desirable to append non-relational metadata to an edge.

For this dissertation relevant types are defined in this or later sections. The following definition for *directed graphs* comes from Bang-Jensen and Gutin [BJG07, p. 2]:

Definition 4.5 (Directed-graph). A **directed graph** (or just **digraph**) D consists of a non-empty finite set $V(D)$ of elements called vertices and a finite set $A(D)$ of ordered pairs of distinct vertices called **arcs**. We call $V(D)$ the **vertex set** and $A(D)$ the **arc set** of D . We will often write $D = (V, A)$ which means that V and

A are the vertex set and the arc set of D , respectively. For an arc (u, v) the first vertex u is its **tail** and the second vertex v is its **head**. We also say that the arc (u, v) **leaves** u and **enters** v . The head and tail of an arc are its **end-vertices**; we say that the end-vertices are **adjacent**, i.e. u is adjacent to v and v is adjacent to u .

So in a digraph the order of the pairs shows the direction. An example for a digraph would be $D = (V, A)$ [BJG07]:

$$V = \{x, y, z, u, v, w\}$$

$$A = \{\{x, z\}, \{y, z\}, \{z, u\}, \{u, v\}, \{u, w\}, \{w, u\}\}$$

Fig. 4.2 shows the corresponding graphical representation to the example digraph D . The definition from Bang-Jensen and Gutin does not allow *multiple* (also called

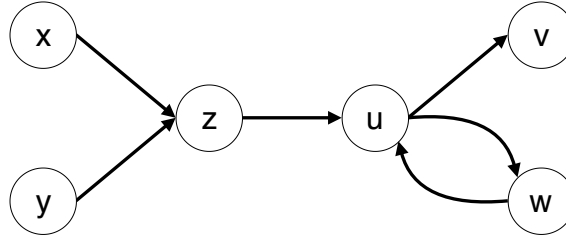


Fig. 4.2: Graphical representation of digraph D

parallel) arcs, which connect the same vertices in the same direction more than once. Also *loops* are not allowed in a directed graph [BJG07, p. 4].

Definition 4.6 (Directed-pseudograph). *When parallel arcs and loops are admissible we speak of **directed pseudographs**; directed pseudographs without loops are **directed multigraphs**.*

In some situations the numbers of vertices and edges is of interest. The mathematical terms used for this are *order* and *size*, as the following definition shows [BJG07, p. 2]:

Definition 4.7 (Size). *The **order (size)** of D is the number of vertices (arcs) in D ; the order of D will be sometimes denoted by $|D|$.*

For the graph D from the example above shown in Fig. 4.2 the order and the size are both 6. Also the number of incoming and outgoing arcs are important. These are defined by different types of degree [BJG07, p. 4]:

Definition 4.8 (In- and out-degree). For a set $W \subseteq V$, the **out-degree** of W (denoted by $d_D^+(W)$) is the number of arcs in D whose tails are in W and heads are in $V - W$, i.e. $d_D^+(W) = |(W, V - W)_D|$. The **in-degree** of W , $d_D^-(W) = |(V - W, W)_D|$. In particular, for a vertex v , the **out-degree** is the number of arcs, except for loops, with tail v . The **degree** of W is the sum of its **out-degree** and **in-degree**.

As for graphs also for digraphs exist subdigraphs with the following definition [BJG07, p. 5]:

Definition 4.9 (Subdigraph). A digraph H is a **subdigraph** of a digraph D if $V(H) \subseteq V(D)$, $A(H) \subseteq A(D)$ and every arc in $A(H)$ has both end-vertices in $V(H)$.

Paths and *cycles*, which are of interest considering learning objective dependencies, are defined as special cases of *walks* [BJG07, p. 10]:

Definition 4.10 (Walk). Let D be a directed pseudograph. A **walk** in D is an alternating sequence $W = x_1 a_1 x_2 a_2 x_3 \dots x_{k-1} a_{k-1} x_k$ of vertices x_i and arcs a_j from D such that the tail of a_i is x_i and the head of a_i is x_{i+1} for every $i = 1, 2, \dots, k-1$. A walk W is **closed** if $x_1 = x_k$, and **open** otherwise. The set of vertices $\{x_1, x_2, \dots, x_k\}$ is denoted by $V(W)$; the set of arcs $\{a_1, a_2, \dots, a_{k-1}\}$ is denoted by $A(W)$. We say that W is a walk **from** x_1 **to** x_k or an (x_1, x_k) -walk. If W is open, then we say that the vertex x_1 is the **initial** vertex of W , the vertex x_k is the **terminal** vertex of W , and x_1 and x_k are **end-vertices**. The **length** of a walk is the number of its arcs.

So based on *walks*, *paths* are defined as follows [BJG07, p. 11]:

Definition 4.11 (Path). If the vertices of the walk W are distinct, W is a **path**. A path P is an $[x, y]$ -**path** if P is a path between x and y , e.g. P is either a (x, y) -path or a (y, x) -path. A **longest path** in D is a path of maximal length in D .

Also based on the definition of *walks* is the definition of *cycles* [BJG07, p. 11]:

Definition 4.12 (Cycle). If the vertices of the walk W are x_1, x_2, \dots, x_{k-1} are distinct, $k \geq 3$ and $x_1 = x_k$, W is a **cycle**. A **longest cycle** in D is a cycle of maximal length in D . When W is a cycle and x is a vertex of W , we say that W is a cycle **through** x . In a directed pseudograph D , a loop is also considered a cycle (of length one).

As *paths* and *cycles* are special cases of *walks* they also inherit their definition of *length*. Because of their properties the *acyclic digraphs* (or directed acyclic graphs) are popular and well-studied. They follow a simple definition [BJG07, p. 13]:

Definition 4.13 (Acyclic-digraph). *A digraph D is **acyclic** if it has no cycle.*

As mentioned, acyclic digraphs have special properties. The following theorems describe a subset of them and have been proven by Bang-Jensen and Gutin [BJG07, p. 13-14]. The proofs will not be part of this dissertation.

Theorem 4.1. *Every acyclic digraph has a vertex of in-degree zero as well as a vertex of out-degree zero.*

Theorem 4.2. *Let D be an acyclic digraph with precisely one vertex x (y) of in-degree (out-degree) zero in D . For every vertex $v \in V(D)$ there is an (x, v) -path and a (v, y) -path in D .*

These theorems support later comparison and analysis.

Furthermore, an important unit of measurement for graphs is the number of *connected components*, which is part of the following definitions.

For the definition of *connected digraphs*, the following additional concepts have to be defined [BJG07, p. 18]:

Definition 4.14 (Biorientation). *For a pseudograph G , a directed pseudograph D is called a **biorientation** of G if D is obtained from G by replacing each edge $\{x, y\}$ of G by either xy or yx or the pair xy and yx (except for a loop xx which is replaced by a (directed) loop at x).*

So, every directed graph can be understood as an biorientation of an undirected graph, which is called *underlying graph* [BJG07, p. 19]:

Definition 4.15 (Underlying graph). *The **underlying graph** $UG(D)$ of a digraph D is the unique graph G such that D is a biorientation of G .*

If every edge of an undirected graph is replaced by an arc in one direction and another arc in the other direction, this biorientation is called complete [BJG07, p. 19]:

Definition 4.16 (Complete biorientation). *For a graph G , the **complete biorientation** of G (denoted by \overleftrightarrow{G}) is a biorientation D of G such that $xy \in A(D)$ implies $yx \in A(D)$.*

With this background knowledge *connected graphs* can be defined, starting with *strongly connected digraphs* [BJG07, p. 16]:

Definition 4.17 (Strongly connected). *A digraph D is **strongly connected** (or, just, **strong**) if, for every pair x, y of distinct vertices in D , there exists an (x, y) -walk and a (y, x) -walk. In other words, D is strong if every vertex of D is reachable from every other vertex of D .*

A *strongly connected digraph* has exactly one *component*, following the definition of a *component* [BJG07, p. 17]:

Definition 4.18 (Strong component). A **strong component** of a digraph D is a maximal induced subdigraph of D which is strong.

With the help of *strongly connected digraphs*, the more general *connected pseudographs* and *connected components* can be defined [BJG07, p. 19]:

Definition 4.19 (Connected). A pseudograph G is **connected** if its complete biorientation \overleftrightarrow{G} is strongly connected. Strong components in \overleftrightarrow{G} are **connected components**, or just **components** in G .

The definition of *connected digraphs* is again more general and requires a connected underlying graph [BJG07, p. 19]:

Definition 4.20 (Connected for digraphs). A digraph is **connected** if its underlying graph is connected.

In contrary to *strongly connected digraphs*, in *connected digraphs* not every vertex has to be reachable from every other vertex. That means, in connected graphs vertices are possible which are not reachable because the directions of the arcs avoid a walk. During the analysis in this dissertation the dependencies which connect two or more connected components are of interest and are called *bridges* [BJG07, p. 19]:

Definition 4.21 (Bridge). A **bridge** in a connected pseudograph G is an edge whose deletion from G makes G disconnected.

The definitions above aim to give a very basic insight into the field of graph-theory and cover all concepts that are needed to understand related work of this and the following chapters. Some of the literature discussed in this chapter is not using mathematical definitions but build its research nonetheless upon graphs. In chapter 6 further related work is discussed also using the graph-theory based definitions.

4.3 Research on Curriculum Analysis

Taking a look over the borders of primary education, a lot of different approaches to analyze and compare curricula are to be found. Most of them focus on academic courses and undergraduate degree programs.

Pedroni, Oriol, and Meyer [POM07] present a framework to model and compare courses and curricula. Based on previous work of Meyer [Mey06] knowledge units and connections between them are used to represent courses and curricula. Knowledge units are distributed into two levels of granularity:

- a **truc** (**Testable, Reusable Unit of Cognition**) is a collection of concepts, operational skills and assessment criteria [Mey06], meeting five characteristics [POM07]:
 1. all components of the truc follow from a central, clearly identified idea;
 2. the truc possesses a clear, unambiguous description;
 3. the truc includes assessment criteria to judge whether a student has mastered the concepts and skills;
 4. included topics span a scope of general interest;
 5. the scope is small enough to be covered in one or a small number of lectures or in a textbook chapter or a few sections

Examples for trucs would be 'object', 'class' or 'feature call'.

- a **notion** represents a single concept or operational skill or a specific facet of a concept and meet three properties [POM07]:
 1. A notion is part of exactly one truc.
 2. It represents a particular view or usage of the trucs to which it belongs.
 3. It may have the same name as the truc to which it belongs, in which case it is said to be the truc's central notion.

Examples for notions, which are part of the example truc 'feature call', would be 'simple qualified feature call' or 'return value'.

Notions can be related using one of the two following relation types:

- **'requires' relation:** notion A **requires** notion B if it is necessary to understand B before A can be understood.
- **'is a' relation:** notion A **is a** notion B if B refines A. With the 'is a' relation notion A inherits all 'requires' relations from B, but may also add additional relations.

'Requires' relations can connect notions of different trucs, so they cross the boundaries of the trucs. In this case the following relation between trucs is defined [POM07]:

- **'dependency' relation:** a truc A depends on a truc B if at least one notion in truc A exists that requires a notion that is part of truc B.

On the contrary 'is a' relations are not allowed to cross borders of a truc. Notions are connected to a *notions graph* using the 'requires' and 'is a' relations as edges and the notions as nodes. As trucs represent sets of notions they are displayed as clusters of notions in a so called *clustered notions graph*, in later publications *truc-notion graph* [PM10]. A *trucs graph* only shows the trucs and their 'dependency' relations and can be abstracted from a *clustered notions graph*. Further a curriculum is defined as a *set of trucs* whereas the notions of these trucs are called the *set of underlying notions* of the curriculum. Courses are understood as a 'list of notions taught one after the other' [POM07], and again the set of all the notions is called *set of underlying notions* of the course. To compare courses or curricula the sets of *set of underlying notions* are used. If one contains the *set of underlying notions* of the other one, they are *compliant*. Additional comparison can show intersections, so which notions courses or curricula have in common, or relative complements, which notions are included in one set but not in the other [POM07]. In later publications Pedroni and Meyer add *cluster* as a type of knowledge unit, representing a collection of trucs and other clusters. With that, *clusters* cover a whole knowledge area [PM10]. With the application of the framework on object-oriented programming, Pedroni and Meyer offer interesting insights in the dependency structure of the graphs. In their analysis, transitive dependencies are considered and the impact of incoming and outgoing links are discussed. Following their opinion, the number of outgoing links can give information about 'a truc's place in a course' [PM10]:

- **trucs with few dependencies** will probably appear towards the beginning
- **trucs with many dependencies** will probably appear towards the end

Further, the number of incoming links represents the number of trucs for which a given node truc is necessary. This can be understood as an indicator for the importance of a truc [PM10]:

- if **many trucs rely on one truc**, then it is probably central to teaching programming and will reappear throughout a course.

Also cycles can occur and add information about the dependencies structure. In the truc and notion graph the circular dependencies indicates that some trucs or notions need to reappear during the learning process in a course or curriculum. Because of this the authors suggest a *spiral model* for teaching object-oriented programming. They base their conclusion on Schwill's *spiral principle for fundamental ideas for computer science* [Sch94], where Schwill applies the ideas of Bruner concerning the process of education to the field of computer science [Bru60, p. 13]:

The early teaching of science, mathematics, social studies, and literature should be designed to teach these subjects with scrupulous intellectual honesty, but with an emphasis upon the intuitive grasp of ideas and

upon the use of these basic ideas. A curriculum as it develops should revisit these basic ideas repeatedly, building upon them until the student has grasped the full formal apparatus that goes with them.

An interesting approach from Germany is Steinert's dissertation 'Lernzielstrukturen im Informatikunterricht' (Learning objective structures in computer science lessons), presenting a detailed analysis method for learning objectives for computer science in school [Ste10]. As an example he uses the curriculum of the mandatory subject informatics in secondary education at Bavarian (Germany) Gymnasia. The steps taken to analyze the curriculum can be described in three major tasks [Ste10]:

1. analysis of the curriculum to identify learning objectives
2. categorization of the learning objectives into the taxonomy of Anderson and Krathwohl [AK01]
3. description of connections between the learning objectives with the help of a learning objectives graph

The first step also include the analysis of examination tasks which should assess if learning objectives were reached. Connected to the tasks the learning objectives are divided into implicit or explicit learning objectives. Explicit learning objectives are those, which are directly included in a task, like 'Develop an algorithm to solve the problem.'. On the other hand implicit learning objectives on the other hand are often invisible in the task, like 'identify relationships' or 'know the UML syntax'. In the second step the learning objectives are categorized into the taxonomy of Anderson and Krathwohl, using the cognitive process and knowledge dimensions described in section 2.5.2. For the connections between the learning objectives Steinert uses the two types of prerequisite relations presented by Hubwieser [Hub08]:

- **'hard' prerequisite** forced by a substantial or logical dependency: a learning objective A is logically necessary for a learning objective B. So B is based on A and B cannot be reached before A is not reached.
- **'soft' prerequisite** suggested by didactically deliberation: a learning objective A is didactically necessary for a learning objective B, but not logical necessary. So B can be reached before A is reached, 'but it is advisable in order to ease or to improve the learning process'.

Both relations are defined to be transitive. In the learning objectives graph these two relations are represented by edges. The following third type of relation, which is also transitive, is displayed in a different way [Ste10].

- **subset relationship:** learning objective B is a partial objective of a learning objective A if the system that describes the knowledge component of learning objective A contains the knowledge component of learning objective B as a component. B is a specialized learning objective of A and A is a generalized learning objective of B. For instance a learning objective containing the object concept in terms of object oriented programming is a generalization of learning objectives containing the concept of attribute or method.

In the learning objectives graph the specialized learning objectives are included in the generalized learning objectives, so they are not displayed as edges of the graph [Hub08, Ste10].

This dissertation uses a comparable approach to represent knowledge units from curricula, educational standards or competency models in primary education. As these educational models are based on standards or competencies, the defined learning outcomes are handled in a similar way as Pedroni and Meyer did it with the trucs [Mey06, POM07, PM10]. In further steps the learning outcomes are related to each other using two comparable types of relationships. This process is comparable to the approach of Hubwieser and Steinert [Hub08, Ste10] but uses different types of relationships and adds metrics for further analysis. As supportive software a graph-database is used to store graph data including attributes of vertices as well as for arcs, combine the single educational models to a generic graph, and calculate measures for analysis. In the following section the representation process is explained.

4.4 A Graph-Based Model

4.4.1 Attributes for Nodes

The graph-based approach developed in this dissertation represents learning objectives, so *competencies* (see section 3.7), of educational models as nodes of directed graphs. The concept of nodes has been defined above but is extended for the approach. This is necessary because the competencies contain more information than a number, a letter, an identification number, or a name. As a minimum an identification number and the text of the competency has to be considered as important attributes. For analysis, comparison, or learning paths generation also the implementation level and the minimum and maximum age range is important. So, the following list of attributes are considered in the approach:

- **name:** is a composition of the country code, a number within the model, and an acronym of the strand and the substrand (e.g. AU-14-DA)
- **text:** the explicit formulation of the competency as it is included in the educational model

- **category:** the overall category used in this approach which the competency can be categorized into (e.g. Digital Infrastructure)
- **subcategory:** the more detailed subcategory which the competency can be categorized into (e.g. Networks)
- **original-category:** the original category from the educational model which the competency is included in (e.g. Computers and Communications Devices)
- **original-subcategory (optional):** the original subcategory from the educational model which the competency is included in (e.g. Human vs. Computers)
- **code (optional):** an identification number of the competency within the original curriculum or standards (e.g. L1:3.CI-1)
- **level:** the level as explained in 3.5 (e.g. level 1)
- **minimum age:** the minimum age of students for which this competency is intended in the original model (e.g. 5 years)
- **maximum age:** the maximum age of students for which this competency is intended in the original model (e.g. 8 years)
- **country:** the country code for the corresponding educational model (e.g. AT)

The *category* (*subcategory*) and the *original-category* (*original-subcategory*) might differ that the former is added by experts and the latter is already specified in the curriculum. As not all models include subcategory it is an optional attribute. The *code* connects the competency directly to a formulation within a curriculum, as some are using identification numbers. It is again optional as some are not using IDs. It can be seen, that the nodes need attributes, but in case of the graph representation the only appearing node type is *competency*. That means, a *vertex-attributed graph* is required.

4.4.2 Types and Attributes for Edges

In this approach the edges of the graph represent added dependencies between the existing learning objectives. As presented by Hubwieser [Hub08] and Steinert [Ste10] also edges have different types. A new feature of this approach is that edges have attributes as well as two different types. The relationships between the competencies are determined by dependencies, in other words prerequisites. Based on the three types of Hubwieser [Hub08] and Steinert [Ste10], *hard* and *soft prerequisites*, and the *subset relationship* (see section 4.3), the here presented approach defines the two relations *expands* and *requires*. The *expands* relation can be compared to the *subset relationship* and is defined as follows:

Definition 4.22 (Expands relation). *A competency $C1$ **expands** a competency $C2$ if $C1$ follows the same central theme as $C2$ does, adding new aspects either to the dimension of breadth or of depth.*

The dimensions *breadth* and *depth* are understood as mentioned in literature, for instance by Schwartz et al. [SSST09]. With *breadth*, or often also *full coverage*, the number of covered topics, which are important for an explicit subject or discipline, is expressed. That means, a focus on *breadth* would lead to a variety of topics. On the other hand, *depth* or *deep coverage* reflects the understanding at different levels. So, focusing on *depth* means focusing on the mastering of fewer fundamental concepts [SSST09]. *Following the same central theme* means that the competencies address the same topic. The following example shows a typical *expands*-relation in the CSTA Computer Science Standards from 2017 [CST17]:

1B-AP-08: *By the end of Grade 5, students will be able to compare and refine multiple algorithms for the same task and determine which is the most appropriate.*

expands

1A-AP-08: *By the end of Grade 2, students will be able to model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.*

In some of the educational models *expands* relations are given explicitly. For example the CSTA Computer Science Standards provide an so called *progress chart* in form of a table. Fig. 4.3 shows an extract from this table.

In the case of the CSTA standards from 2017 the progression chart is given for four levels. That means, the two levels shown in Fig. 4.3 are followed by two additional levels which can be seen in Fig. 4.4.

The categories again contain subcategories and for each of them standards for different levels are described. Standards, which are in the same subcategory (row) but in different levels (column), follow the same *central theme*. So, each standard in a higher level *expands* the standard in the previous level. To preserve the information from the models, the *expands* relation is defined. In those educational models, where no information about the order of competencies is given, *expands* relations are added, following the given definition.

The *requires* relation is comparable to *hard prerequisites* and defined as follows:

Definition 4.23 (Requires relation). *A competency $C1$ **requires** a competency $C2$ if $C1$ cannot be reached without reaching $C2$ before and $C1$ does not **expand** $C2$.*

An example for a *requires*-relation can be seen between the two competencies from the CSTA Computer Science Standards from 2011 [SCF⁺11]:

Concept	Subconcept	Level 1A (Ages 5-7)	Level 1B (Ages 8-11)
		By the end of Grade 2, students will be able to...	By the end of Grade 5, students will be able to...
Computing Systems	Devices	1A-CS-01 Select and operate appropriate software to perform a variety of tasks, and recognize that users have different needs and preferences for the technology they use. (P1.1)	1B-CS-01 Describe how internal and external parts of computing devices function to form a system. (P7.2)
	Hardware & Software	1A-CS-02 Use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware). (P7.2)	1B-CS-02 Model how computer hardware and software work together as a system to accomplish tasks. (P4.4)
	Troubleshooting	1A-CS-03 Describe basic hardware and software problems using accurate terminology. (P6.2, P7.2)	1B-CS-03 Determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies. (P6.2)
Networks & The Internet	Network Communication & Organization		1B-NI-04 Model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination. (P4.4)
	Cybersecurity	1A-NI-04 Explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access. (P7.3)	1B-NI-05 Discuss real-world cybersecurity problems and how personal information can be protected. (P3.1)

Fig. 4.3: The first rows for the first two levels of the progression chart for the CSTA Standards 2017 [CST17]

L1:3.CL.1: *The students will be able to gather information and communicate electronically with others with support from teachers, family members or student partners.*

requires

L1:3.CD.1: *The students will be able to use standard input and output devices to successfully operate computers and related technologies.*

Students, that are not able to handle digital devices, in this case especially the computer, would not be able to use them for information collection or communication. This dependency is pretty clear, although the two competencies are part of different categories: in the original standards **L1:3.CL.1** is located in the category *Collaboration* and the subcategory *Using technology tools and resources for collaboration*, where **L1:3.CD.1** belongs to the category *Computers and Communications Devices* and the subcategory *Computers* [SCF⁺11]. So, **L1:3.CL.1** and **L1:3.CD.1** are not part of the same category and do not follow the same central theme.

The point is, that the *expands*-relation is a special case of the *requires*-relation. So, an *expands*-relation contains the information, that the two related competencies follow the same central theme. One reason to use these two different types of relations is not to lose additional information given by the curricula or standards documents.

Besides the types, relations also include a set of attributes to store additional

Level 2 (Ages 11-14)	Level 3A (Ages 14-16)
By the end of Grade 8, students will be able to...	By the end of Grade 10, students will be able to...
2-CS-01 Recommend improvements to the design of computing devices, based on an analysis of how users interact with the devices. (P3.3)	3A-CS-01 Explain how abstractions hide the underlying implementation details of computing systems embedded in everyday objects. (P4.1)
2-CS-02 Design projects that combine hardware and software components to collect and exchange data. (P5.1)	3A-CS-02 Compare levels of abstraction and interactions between application software, system software, and hardware layers. (P4.1)
2-CS-03 Systematically identify and fix problems with computing devices and their components. (P6.2)	3A-CS-03 Develop guidelines that convey systematic troubleshooting strategies that others can use to identify and fix errors. (P6.2)
2-NI-04 Model the role of protocols in transmitting data across networks and the Internet. (P4.4)	3A-NI-04 Evaluate the scalability and reliability of networks, by describing the relationship between routers, switches, servers, topology, and addressing. (P4.1)
2-NI-05 Explain how physical and digital security measures protect electronic information. (P7.2)	3A-NI-05 Give examples to illustrate how sensitive data can be affected by malware and other attacks. (P7.2)
2-NI-06 Apply multiple methods of encryption to model the secure transmission of information. (P4.4)	3A-NI-06 Recommend security measures to address various scenarios based on factors such as efficiency, feasibility, and ethical impacts. (P3.3)
	3A-NI-07 Compare various security measures, considering tradeoffs between the usability and security of a computing system. (P6.3)
	3A-NI-08 Explain tradeoffs when selecting and implementing cybersecurity recommendations. (P7.2)

Fig. 4.4: The first rows for the third and fourth level of the progression chart for the CSTA Standards 2017 [CST17]

information about e.g. their evaluation. Important aspects for this dissertation are reflected by the following attributes:

- **added [boolean]**: a simple attribute with the possible values *true* and *false*. It provides information, if the relation derives from the original model (not added, therefore the value is *false*) or has been added by experts (value *true*).
- **similarity [float]**: contains a calculated number which reflects on the linguistic similarity. How the values of this attribute are determined is discussed in *Part 3: Implementation*.
- **status [string]**: this attribute either contains the value *tbe* (stands for *to be evaluated*), *ok*, or *notok*. It adds information, if a competency has been evaluated by at least one expert or not, and if the relation is considered to be correct or not. If more than one expert evaluated the relation, the majority decides if the status is set to *ok* or *notok*.
- **suggestion [string]**: contains one or more suggestions about what should be changed, if the relation is evaluated to be not correct.

- **prerequisitefactor [string]:** this factor can contain the values *hard* and *soft* and adds information about the degree of the dependency. A *hard* factor shows an absolute mandatory prerequisite, whereas a *soft* factor shows an optional prerequisite. This attribute is not used in this dissertation but already added for future extensions.

The attributes for the relations opens a lot of further opportunities to store information about the transition from one competency to the other. These can for instance be a *complexity factor*, an *estimated duration*, or an *importance value*. All of them would be of great interest for research or education, but need a lot of time for testing and evaluating. Due to page and time limits, they are considered for future work, but are not part of this dissertation.

Both relation types are of course *transitive*. That means, if a competency C3 *expands* the competency C2 and C2 *expands* the competency C1, then C3 *expands* also C1. The same implication is valid for *requires*. It gets more complex, when the transitivity over two different relations types is considered. This can happen in two ways: C3 *requires* C2 and C2 *expands* C1, or C3 *requires* C2 and C2 *expands* C1. In the first case, shown in Fig. 4.5, it seems clear, that between C3 and C1 a transitive *requires*-relation exists. With C2 and C3 related over a *requires*-relation, the central theme from C2 is not continued in C3. On the other hand, the central theme from C1 is continued by C2, displayed by their *expands*-relation. For this reason, C1 and C3 do not follow the same central theme, although C1 is still necessary for C3.

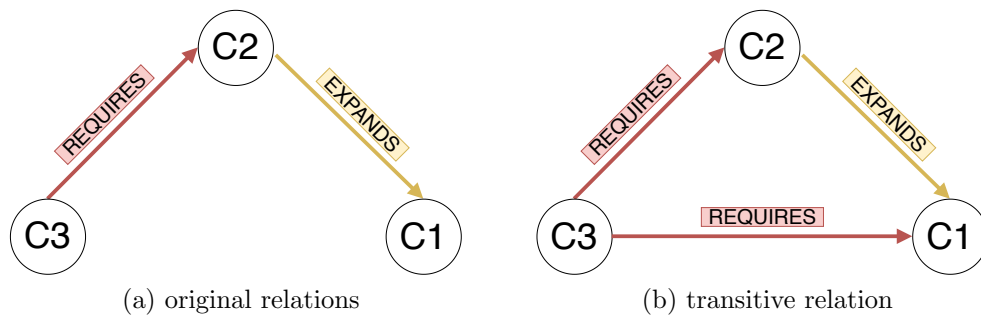


Fig. 4.5: Transitivity in the case of a first *requires*-relation followed by an *expands*-relation

For the second case, shown in Fig. 4.6, also a transitive *requires*-relation between C3 and C1 is considered. As C3 *expands* C2, it follows the same central theme. C2 is related to C1 over a *requires*-relation. This means, that C2 and C1 don't follow the same central theme. As C3 continues the central theme of C2, it can not follow the central theme of C1. Therefore, C3 *requires* on C1.

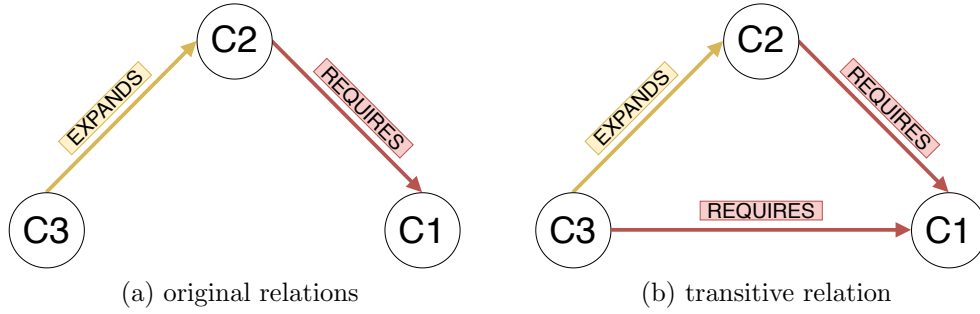


Fig. 4.6: Transitivity in the case of a first *expands*-relation followed by an *requires*-relation

For complexity and readability concerns transitive relations are not shown in the graphs. Otherwise the graphs would contain an extensive number of relations, which would make the graph unreadable. Further, it would impact analysis and measures.

To represent all the desired information an *edge-labeled* and *edge-attributed graph* is necessary.

4.4.3 Property Graph

The described requirements constitute a step towards the field of *data structures* and *data management*, where the concept of so called *property graphs* exists and finds application [RWE15]. Originally the notion of this graph was introduced by Rodriguez and Neubauer [RN10] and recently formalized by Angles [Ang18]. It is based on the following assumptions [Ang18, p. 2]:

- L is a finite set of labels (for nodes and edges);
- P is an infinite set of property names;
- V is an infinite set of atomic values, and its values are distinguished by quoted strings;
- T is a finite set of datatypes (e.g., integer);
- $SET^+(X)$, with a given set X , is the set of all finite subsets of X , excluding the empty set.

The definition of a *property graph* is presented by Angles as follows [Ang18, p. 2]:

Definition 4.24 (Property graph). A **property graph** is a tuple of $G = (N, E, \rho, \lambda, \sigma)$ where:

1. N is a finite set of nodes (also called vertices);
2. E is a finite set of edges such that E has no elements in common with N ;
3. $\rho : E \rightarrow (N \times N)$ is a total function that associates each edge in E with a pair of nodes in N (i.e. ρ is the usual incidence function in graph theory);
4. $\lambda : (N \cup E) \rightarrow SET^+(\mathbf{L})$ is a partial function that associates a node/edge with a set of labels from \mathbf{L} (i.e. λ is a labeling function for nodes and edges);
5. $\sigma : (N \cup E) \times \mathbf{P} \rightarrow SET^+(\mathbf{V})$ is a partial function that associates nodes/edges with properties, and for each property it assigns a set of values from \mathbf{V} .

If two given nodes $n_1, n_2 \in N$ are connected over an edge $e \in E$, such that $\rho(e) = (n_1, n_2)$, node n_1 is called 'source node' and n_2 'target node' of e . A property is given by $(o, p) \in (N \cup E) \times P$ and its assignment $\sigma(o, p) = \{v_1, \dots, v_n\}$, and $(o, p) = v_i$ with $1 \leq i \leq n$ represents the shorthand for a single property. In this case o is the 'property owner', p is the 'property name' and v is the 'property value' [Ang18].

An example subgraph from the CSTA Computer Science Standards from 2011 [SCF⁺11], limited to three nodes, is shown in Fig. 4.7, including the following three competencies:

- **US-1-DA:** The students will be able to gather information and communicate electronically with others with support from teachers, family members or student partners.
- **US-14-DI:** The students will be able to use standard input and output devices to successfully operate computers and related technologies.
- **US-17-DA:** The students will be able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual and collaborative writing, communication and publishing activities.

The graph in Fig. 4.7 is formally described as follows:

$$\begin{aligned}
 N &= \{n_1, n_2, n_3\} \\
 E &= \{e_1, e_2\} \\
 \lambda(n_1) &= \text{Competency}, \\
 (n_1, \text{name}) &= \text{'US-1-DA'}, \\
 (n_1, \text{text}) &= \text{'The students will be able to gather information and} \\
 &\quad \text{communicate electronically with others with support from} \\
 &\quad \text{teachers, family members or student partners.'}, \\
 (n_1, \text{category}) &= \text{'Digital Applications'},
 \end{aligned}$$

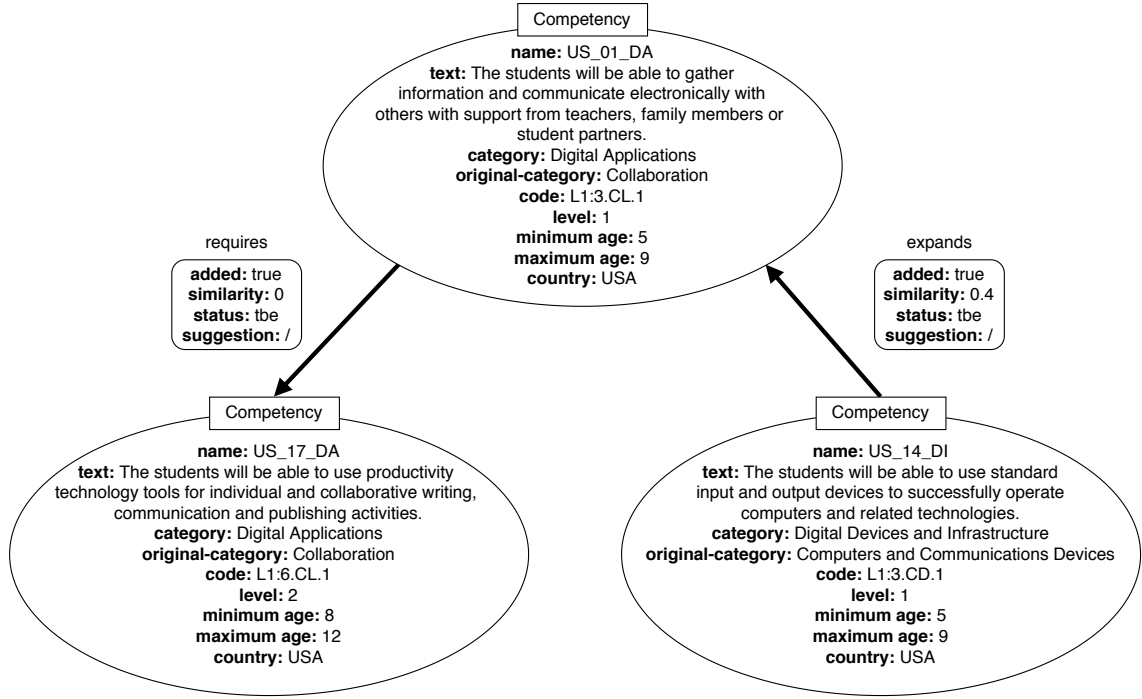


Fig. 4.7: An example of a property graph using the properties for competencies and the two relation-types

$$\begin{aligned}
 (n_1, original - category) &= 'Collaboration', \\
 (n_1, code) &= 'L1:3.CL.1', \\
 (n_1, level) &= '1'; \\
 (n_1, minimum - age) &= '5'; \\
 (n_1, maximum - age) &= '9'; \\
 (n_1, country) &= 'USA'
 \end{aligned}$$

$$\begin{aligned}
 \lambda(n_2) &= \text{Competency}, \\
 (n_2, name) &= 'US-14-DI', \\
 (n_2, text) &= 'The students will be able to use standard input and \\
 &\quad \text{output devices to successfully operate computers and} \\
 &\quad \text{related technologies.}', \\
 (n_2, category) &= 'Digital Devices and Infrastructure', \\
 (n_2, original - category) &= 'Computers and Communications \\
 &\quad \text{Devices}', \\
 (n_2, code) &= 'L1:3.CD.1', \\
 (n_2, level) &= '1'; \\
 (n_2, minimum - age) &= '5';
 \end{aligned}$$

$$\begin{aligned}(n_2, \textit{maximum} - \textit{age}) &= '9'; \\ (n_2, \textit{country}) &= 'USA'\end{aligned}$$

$$\begin{aligned}\lambda(n_3) &= \textit{Competency}, \\ (n_3, \textit{name}) &= 'US-17-DA', \\ (n_3, \textit{text}) &= 'The students will be able to use productivity \\ &\quad \textit{technology tools for individual and collaborative writing,} \\ &\quad \textit{communication and publishing activities.}', \\ (n_3, \textit{category}) &= 'Digital Applications', \\ (n_3, \textit{original} - \textit{category}) &= 'Collaboration', \\ (n_3, \textit{code}) &= 'L1:6.CL.1', \\ (n_3, \textit{level}) &= '2'; \\ (n_3, \textit{minimum} - \textit{age}) &= '8'; \\ (n_3, \textit{maximum} - \textit{age}) &= '12'; \\ (n_3, \textit{country}) &= 'USA'\end{aligned}$$

$$\begin{aligned}\rho(e_1) &= (n_1, n_2), \lambda(e_1) = \{\textit{requires}\}, \\ (e_1, \textit{added}) &= 'yes', \\ (e_1, \textit{similarity}) &= '0', \\ (e_1, \textit{status}) &= 'tbe', \\ (e_1, \textit{suggestion}) &= ' '\end{aligned}$$

$$\begin{aligned}\rho(e_2) &= (n_3, n_1), \lambda(e_2) = \{\textit{expands}\}, \\ (e_2, \textit{added}) &= 'no', \\ (e_2, \textit{similarity}) &= '0.4', \\ (e_2, \textit{status}) &= 'tbe', \\ (e_2, \textit{suggestion}) &= ' '\end{aligned}$$

Angels additionally defines *integrity constraints* for property graphs. Referencing to Codd [Cod80], he describes them as 'general statements and rules that define the set of consistent database states or changes of a state or both' [Ang18, p. 3]. Concerning property graphs Angels focuses on *data schemas*, in this case *property graph schemas*, to specify the types of nodes, the types of edges, and the properties each type can contain. The *property graph schemas* are defined as follows [Ang18, p. 4]:

Definition 4.25 (Property graph schema). *A **property graph schema** is a tuple $S = (T_N, T_E, \beta, \gamma)$ where:*

1. $T_N \subset L$ is a finite set of labels representing node types;

2. $T_E \subset L$ is a finite set of labels representing edge types, satisfying that T_E and T_N are disjoint;
3. $\beta : (T_N \cup T_E) \times P \rightarrow T$ is a partial function that defines the properties for node and edge types, and the datatypes of the corresponding values;
4. $\delta : (T_N, T_N) \rightarrow SET^+(T_E)$ is a partial function that defines the edge types allowed between a given pair of node types.

Fig. 4.8 shows the property graph schema used in this approach for the competency graphs.

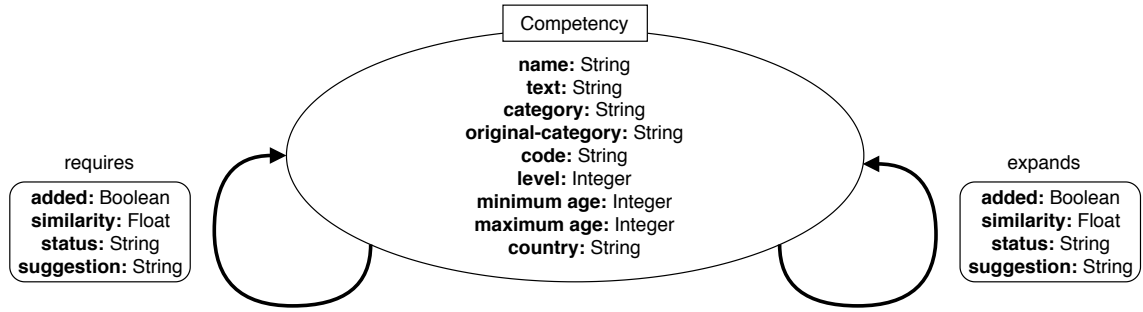


Fig. 4.8: The graph schema for the competency graphs

The formal description for the schema shown in Fig. 4.8 looks like follows:

$$\begin{aligned}
 T_N &= \{Competency\} \\
 T_E &= \{expands, requires\} \\
 \beta(Competency, name) &= String \\
 \beta(Competency, text) &= String \\
 \beta(Competency, category) &= String \\
 \beta(Competency, originalcategory) &= String \\
 \beta(Competency, code) &= String \\
 \beta(Competency, level) &= Integer \\
 \beta(Competency, minimumage) &= Integer \\
 \beta(Competency, maximumage) &= Integer \\
 \beta(Competency, country) &= String \\
 \beta(expands, added) &= Boolean \\
 \beta(expands, similarity) &= Float \\
 \beta(expands, status) &= String \\
 \beta(expands, suggestion) &= String \\
 \beta(requires, added) &= Boolean \\
 \beta(requires, similarity) &= Float \\
 \beta(requires, status) &= String
 \end{aligned}$$

$$\begin{aligned}
\beta(\text{requires}, \text{suggestion}) &= \text{String} \\
\delta(\text{Competency}, \text{Competency}) &= \{\text{expands}\} \\
\delta(\text{Competency}, \text{Competency}) &= \{\text{requires}\}
\end{aligned}$$

Angles further defines that a *schema-instance consistency* is given, when an instance property graph satisfies the restrictions specified in a property graph schema, and defined as follows [Ang18, p. 4]:

Definition 4.26 (Consistency of a property graph schema). *Given a property graph $G = (N, E, \rho, \lambda, \sigma)$ and a property graph schema $S = (T_N, T_E, \beta, \delta)$, we say that G is valid with respect to S when:*

1. *For each node $n \in N$, it applies that $\lambda(n) \subseteq T_N$;*
2. *For each edge $e \in E$, it applies that $\lambda(e) \subseteq T_E$;*
3. *For each node or edge property $(o, p) = v$ in G , it satisfies that there exists $\beta(t_o, p) = dt$ in S such that $t_o \in \lambda(o)$ and $\text{type}(v) = dt$;*
4. *For each edge $e \in E$ such that $\rho(e) = (n, n')$, it satisfies that there exist $l_e \in \delta(t, t')$ such that $l_e \in \lambda(e)$, $t \in \lambda(n)$ and $t' \in \lambda(n')$.*

With a property graph G and a property graph schema S a *graph database* can be defined as the tuple $D(S, G)$, and D satisfies the constraints from the schema-instance consistency if G is valid with respect to S [Ang18, p. 5]. What *graph databases* are, which one is used and which benefit it derives for this dissertation is described in the following section.

4.4.4 Graph Database

As seen in the previous section the notion of a *property graph* leads to a realization with a *graph database*. *Graph databases* belong to the family of NoSQL databases, which differ in some aspects from traditional relational databases. One of the main reasons for the development of NoSQL databases is that SQL-based systems become harder to use (Robinson et al. call it *unwieldy* [RWE15]) handling large datasets. Further, the data changes faster and the structure varies more often. So, NoSQL databases were developed to tackle these weaknesses of relational databases, by the introduction of alternative models, like for instance graph based models. *Graph databases* improve the performance, flexibility and agility in the handling of connected data compared to relational databases. These improvements are possible, because some graph databases use a *native graph storage* to store data. Nodes and relations are directly stored as nodes and relations and do not rely on indexing [RWE15].

```

1 START n=node(*), n2=node(*)
2 MATCH (n {organization: 'CSTA'})
3 OPTIONAL MATCH (n)-[r]-(n2)\
4     WHERE n2.organization = 'CSTA'
5 RETURN count(DISTINCT n) as nrNodes,
6     count(DISTINCT r) as nrEdges,
7     count(DISTINCT r) / ((count(DISTINCT n)) * (count(DISTINCT n)
    - 1.0)) AS graphDensity

```

Listing 4.1: Query for the density of a graph

A graph database that is based on property graphs is *neo4j* [Neo]. It brings its own query language called *cypher*. Listing 4.1 is written in *cypher*, based on a query published online¹ and has already been published by Pasterk and Bollin [PB17b]. It shows one example of the many queries that were used in this research and calculates the density of the graph of the CSTA Computer Science Standards from 2011.

The graph database offers an interesting platform to represent curricula in form of graphs and to analyze and compare different educational models. For the approach of this dissertation the graph database *neo4j* is used to store the data, model the graph, and calculate the needed graph-theoretic metrics. It has to be mentioned, that some of the calculations done for this approach with *neo4j* are mentioned by Van Bruggen to be 'not to use graph database'-cases [Bru14]. As there are more 'to use graph database'-cases in the approach, *neo4j* was selected to be the basic platform. In further steps it is used to build an online service to develop own curricula in the field of computer science education. The steps done to represent the educational models in a graph database are described in the next section.

4.5 Generating the Graphs

4.5.1 Identification of Comparable Elements

As described in section 2.2, a curriculum as well as educational standards or competency models consist of different elements. The question, which of these elements are most comparable, seems to be valid at this point. Taking a look at the work of Glatthorn et al., and the five already listed components, it can be argued, which elements are of interest for a detailed comparison [GBWB15, p. 6]:

- **A rational for the curriculum:** the rational depends on cultural aspects of the particular countries. To compare these components, a cultural analysis of

¹ Cypher query to calculate the density:
<http://www.makedatauseful.com/neo4j-cypher-query-for-graph-density-analysis/>

the countries is necessary. Further, the results achieved by the comparison of rationals for curricula is very limited.

- **The aims, objectives, and content for achieving those objectives:** the aims, objectives and content include a lot of information. They reflect the national view of the subject including only the most important aspects of the area. In most cases the aims, objectives and content deliver some kind of expert opinion or knowledge, as they are developed by groups of experts. Further these elements represent important skills and the direction students are guided in. Teachers need this information in order to be able to plan and hold their lessons accordingly. So, the aims, objectives and content provide teachers some kind of framework to work with.
- **Instructional methods:** a comparison of the methods is of great interest. But it has to be considered that they are generally applicable in most cases and do not necessarily reflect the characteristic of a subject. That means similar instructional methods are applied in different subjects. A comparison of the methods used in one subject is only a limited view. Comparing the methods applied in different scientific areas or subjects would be of more interest.
- **Learning materials and resources:** this component varies from case to case. Some models provide very detailed material, others include limited resources, and others contain no material at all. If material is included, it appears in different forms. Therefore, a comparison of this component is not appropriate.
- **Tests or assessment methods:** similar arguments can be found for this component as for the instructional methods. Tests and assessment methods are not necessarily characteristic for a subject, as they can be applied for several subjects. Nevertheless, a comparison would be of great interest. Especially comparing the methods of different subjects again.

The comparison of several components is of interest. But following these arguments, the component which reflects the characteristics of a subject best, are *the aims, objectives, and content*. During discussions with teachers and researchers active in the field of computer science education, also learning objectives were identified as important parts of curricula and worthwhile a detailed analysis and comparison. Based on these arguments the component of *the aims, objectives, and content* are selected for further research and carved out from the models to be represented as nodes in the graphs.

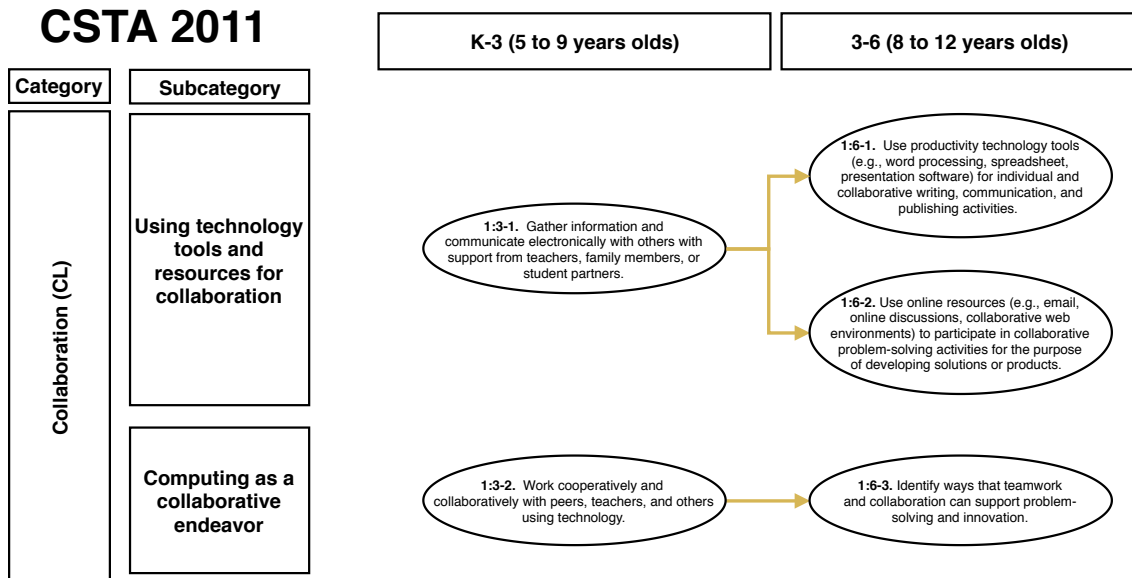


Fig. 4.9: CSTA 2011 category 'Collaboration' showing dependencies within the category

4.5.2 Creation of Relations

The relations were added in two phases, one for *expands*- and one for *requires*-relations. In the first step of phase one the *expands*-relations given by the curricula were directly transferred into the yet empty graphs. This can be seen in Fig. 4.9, showing the category 'Collaboration' from the CSTA Standards 2011 with the two subcategories 'Using technology tools and resources for collaboration' and 'Computing as a collaborative endeavor'. The levels K-3 and 3-6 are represented as columns. The yellow arrows display the given *expands*-relations. It can be seen that the relations are limited to a category and in most cases also to a subcategory. If no sequence was included in the corresponding curriculum, *expands*-relations were added, considering the definition for this type of relations (see definition 4.22) as well as the categories. In some cases also *expands*-relations crossed the borders of one category, because of related topics. This action concluded phase one.

During phase two the *requires*-relations were added, which in most cases are not limited to one category. Again it depended on the definition of the *requires*-relation (see definition 4.23), if a relation was added or not. Fig. 4.10 again shows the given *expands*-relations as yellow arrows and the added *requires*-relations as red arrows. As it can be seen, only one of the *requires*-relations is located in the category 'Collaboration', crossing from one subcategory to the other one. The remaining *requires*-relations come from or point into four different categories and six subcategories.

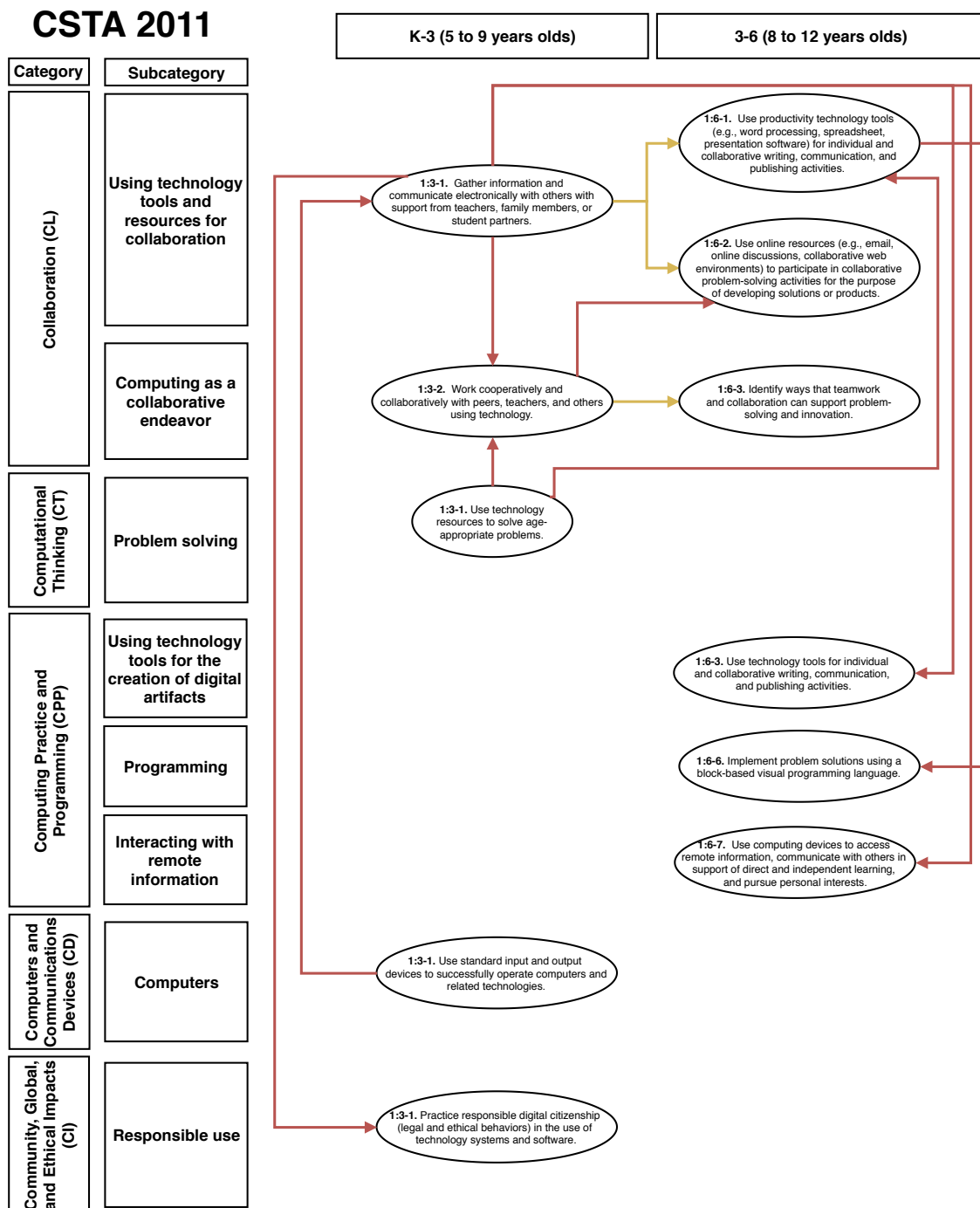


Fig. 4.10: CSTA 2011 category 'Collaboration' including dependencies to other categories

Both relation types were generated by the author for all educational models. For evaluation three experts reviewed the relations and afterwards they were revised. This process is described in the following section.

4.5.3 Reviews and Revision

Three experts, all researchers in the field of computer science education, reviewed the generated relations to evaluate if they are convenient and if they follow the given definitions. Two experts evaluated the relations by reading the related competencies in form of tables and choose, if they agree or disagree with the given relation. They had also the possibility to add comments or suggest other relations. The third expert had the opportunity to use and test the GECKO platform, a platform developed for this kind of evaluation and presented in chapter 7. In GECKO the expert had the same task but in a more user-friendly environment and was directly working on the graphs instead of tables.

Taking a look at the results from the experts' evaluation, an average *agree-disagree-ratio* of three to one (three agree to one disagree) is visible. So, just in a quarter of all existing relationships, the experts disagreed. Reasons for a '*disagree*' a '*wrong direction*', a '*wrong type*' or '*no dependency*' were equally frequently mentioned. From all relations evaluated the percentage of disagree decisions was 24%. The opinions differed especially in the curriculum from Australia, with the highest number of disagree decisions (27%). The highest agreement can be found in the curriculum from Switzerland with a percentage of 21% of disagree decisions. In the results no explicit difference between the experts working on tables or on the GECKO platform could be found.

Two mayor points from the feedback of the experts have to be considered. First, although the relations were clearly defined, the results of this process lead to broad and long-term discussions. The experts' opinions differed in several aspects, like in which direction a relation points or if a dependency between two competencies is necessary or not. Even the types of relations induced discussions. This was of course influenced by prior experiences or preferences of each expert. Second, all the experts reported, that it was very hard to remember all the competencies when they tried to find missing relations. The evaluation of the existing relations were discussed, but they were not sure, if they missed a necessary relation. This was feedback by the two experts handling the tables as well as the one working on the GECKO platform. These two points lead to a technical and semi-automated approach based on natural language processing, which is described in chapter 8. The tools developed for this approach support the experts by suggestions for additional relations, based on the linguistic similarity of the competencies.

The majority of experts opinions decided, if changes had to be made in the original model. So, if at least two experts evaluated a relation as '*wrong direction*'

or '*wrong type*', the direction or the type was changed. If two experts chose '*no dependency*' for a relation, it was deleted.

The reviews and the revision represented the last steps of the single curriculum graph-generating process. After this each curriculum was represented as a graph to enable further analysis and comparison. In section 5 the methods and results of this analysis and comparison are presented. As a next step, the graph-based models for each single curriculum are combined to a *Generic, Graph-Based Model for Competencies (GGBMC)*. In the following section the necessary steps in the process towards the GGBMC are described.

4.6 Generic, Graph-Based Model for Competencies (GGBMC)

4.6.1 Steps in the Process

The goal of the *Generic, Graph-Based Model for Competencies (GGBMC)* is to combine all graphs from the curricula to one generic graph. For this, the following three steps of preliminary work are necessary.

1. **Standardization:** as some of the competency formulations differ in content and detail, a standardization is necessary before the curricula can be combined. For this, linguistic elements like *commas*, '*and*' and '*or*' are used to split formulations which contain more than one predicate or object.
2. **Categorization:** to identify similar competencies, they are categorized into the category system for computer science education presented by Duncan and Bell [DB15] as described in section 3.2.
3. **Combination:** to combine the curricula to one graph, similar competencies are identified and used as connection elements.

These steps display major milestones within the process and also consist of a number of smaller activities. It has to be mentioned that the order of the steps plays an important role. The first two steps are necessary for step three. Step one is necessary for step two because it can occur that split formulations can be categorized into different categories, although they are part of the same original formulation. The three steps are described in more detail in the following three sections.

4.6.2 Standardization of Competencies

In a first step of standardization, all competency formulations are reformulated into a form with the prefix '*students are able to*'. Most of the selected models included

formulations of that kind, except the Austrian digikomp [Dig13b] and the model from Berry for the English curriculum [Ber15] using 'I can' statements.

One major issue in combining the curricula to one model are the different formulations of competencies within the curricula. Some formulations use one *predicate*, that explains what students should be able to do with one *object*. Others describe multiple actions with more than one *predicate* for one *object*. Still others contain one *predicate* for multiple *objects*. Some list several actions for several *objects*. It also appears, that one formulation includes two sentences. In the following part an example for each of these forms is presented:

- An example for one predicate and one object from the CSTA Standards of 2017 [CST17]:

Students are able to decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

- An example for multiple predicates and one object from CSTA Standards of 2011 [SCF⁺11]:

Students are able to understand and use the basic steps in algorithmic problem-solving.

- An example for one predicate and multiple objects from the Standards of the German Informatics Society [Ges19]:

Students are able to describe states and state transitions of automata.

- An example for multiple predicates and multiple objects from the curriculum from Switzerland [Leh14]:

Students are able to recognize and use tree and network structures.

- An example for two sentences in one formulation from the Australian Curriculum [Aus13], also including multiple predicates and multiple objects:

Students are able to create and organise ideas and information using information systems, and share information in safe online environments.

These given examples are clear representatives of the different forms. But there are a lot more complex formulations, like the following examples show:

- from the standards of the German Informatics Society [Ges19]:

Students are able to design, implement and test algorithms with the basic algorithmic building blocks instruction, sequence, repetition and branching.

- from the CSTA Standards of 2017 [CST17]:

Students are able to modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

- from the CSTA Standards of 2017 [CST17]:

Students are able to take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development.

All curricula contain formulations in every form. To standardize them, an approach applied by Hubwieser et al. [HGB⁺15] is used. They split statements with 'and' respective 'or' operators into two statements. For this dissertation the method is extended to also split statements with *commas*, which appear in formulations with multiple predicates or objects and some *dependent clauses*. *Commas* between multiple predicates, as well as objects, can be handled as simple sentence splits. *Dependent clauses* have to be treated individually, as in most cases they display a more complex structure. The statements resulting from the division are called *subcompetencies* because definition 2.18 applies to them. So, for the representative examples the standardized *subcompetencies* are as follows:

- One predicate and one object [CST17]:

Students are able to decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

No *subcompetencies*.

- Multiple predicates and one object [SCF⁺11]:

Students are able to understand and use the basic steps in algorithmic problem-solving.

Subcompetencies:

- Students are able to understand the basic steps in algorithmic problem-solving.
- Students are able to use the basic steps in algorithmic problem-solving.

- One predicate and multiple objects [Ges19]:

Students are able to describe states and state transitions of automata.

Subcompetencies:

- Students are able to describe states of automata.
- Students are able to describe state transitions of automata.

- Multiple predicates and multiple objects [Leh14]:

Students are able to recognize and use tree and network structures.

Subcompetencies:

- Students are able to recognize tree structures.
- Students are able to recognize network structures.
- Students are able to use tree structures.
- Students are able to use network structures.

- Two sentences in one formulation [Aus13], also including multiple predicates and multiple objects:

Students are able to create and organise ideas and information using information systems, and share information in safe online environments.

Subcompetencies:

- Students are able to create ideas using information systems.
- Students are able to create information using information systems.
- Students are able to organise ideas using information systems.
- Students are able to organise information using information systems.
- Students are able to share information in safe online environments.

For the examples and statements in this form the procedure is clear. More complex formulations are also more complex to split. A lot of cases need an individual treatment and result in a large number of *subcompetencies*, as the following example from the Standards of the German Informatics Society [Ges19] shows.

Students are able to design, implement and test algorithms with the basic algorithmic building blocks instruction, sequence, repetition and branching.

Subcompetencies:

- Students are able to design algorithms with the basic algorithmic building blocks instruction.
- Students are able to design algorithms with the basic algorithmic building blocks sequence.
- Students are able to design algorithms with the basic algorithmic building blocks repetition.
- Students are able to design algorithms with the basic algorithmic building blocks branching.
- Students are able to implement algorithms with the basic algorithmic building blocks instruction.
- Students are able to implement algorithms with the basic algorithmic building blocks sequence.
- Students are able to implement algorithms with the basic algorithmic building blocks repetition.
- Students are able to implement algorithms with the basic algorithmic building blocks branching.
- Students are able to test algorithms with the basic algorithmic building blocks instruction.
- Students are able to test algorithms with the basic algorithmic building blocks sequence.
- Students are able to test algorithms with the basic algorithmic building blocks repetition.
- Students are able to test algorithms with the basic algorithmic building blocks branching.

In this case there are not more objects but variations of it defined by its components. That makes it necessary to split, although it is more or less the same object. Other formulations contain explanations of what something is used for. Such cases become even more difficult, if there are more than one use cases described, like the following example from the CSTA Standards of 2017 [CST17] shows.

Students are able to modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

Subcompetencies:

- Students are able to modify portions of an existing program into one's own work, to develop something new.
- Students are able to remix portions of an existing program into one's own work, to develop something new.
- Students are able to incorporate portions of an existing program into one's own work, to develop something new.
- Students are able to modify portions of an existing program into one's own work, to add more advanced features.
- Students are able to remix portions of an existing program into one's own work, to add more advanced features.
- Students are able to incorporate portions of an existing program into one's own work, to add more advanced features.

Here, the goals influence the object, because it makes a difference with which goal a program portion is modified, remixed and finally incorporated. That makes a split necessary. Another complex example comes again from the CSTA Standards of 2017 [CST17]:

Students are able to take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development.

Subcompetencies:

- Students are able to take on varying roles, with teacher guidance, when collaborating with peers during the design stages of program development.
- Students are able to take on varying roles, with teacher guidance, when collaborating with peers during the implementation stages of program development.
- Students are able to take on varying roles, with teacher guidance, when collaborating with peers during the review stages of program development.

This last example shows a dependent clause that describes different stages in which students should be able to take roles. The three stages make a split necessary, as they change the object by the different roles in the different stages.

As a result of this steps of standardization the number of elements in each curriculum rose (see the numbers in section 5). In the graphs of the curriculum some nodes representing the competencies stayed the same and some others were substituted by new nodes representing the subcompetencies. A competency with more than one subcompetency was replaced by all of its subcompetencies. This of course had also impact on the relations between the competencies. Within the set of the subcompetencies of one competency the order of the original formulation was preserved, as it can be seen in the examples. Further all of the subcompetencies of one competency were connected over an *expands-relation*, because they follow the same topic and add new aspects. All the dependencies to other competencies were transferred to the first subcompetency and all dependencies from other competencies pointed to the last of the subcompetencies in the sequence. In an additional revision of the relations between the standardized competencies, also some new *requires-relations* were added, if it was necessary.

The incorporation of the standardized competencies into the graphs marked the end of this step. In further analysis the standardized competencies are used most of the time. Additionally, the comparison of the single curricula, described in section 5.4, is done with original as well as standardized competencies. The semi-automated processes, described in part 3, are based on the standardized competencies. As the next step the standardized competencies were categorized into well known computer science areas for education. In the following section this process is described in detail.

4.6.3 Categorization of Competencies

The process of categorization was again divided into two major steps applying different methods. At first an overview of the general focus of the curricula was of interest. For this purpose the two categories '*computer science*' and '*digital literacy*' were selected, as they are suggested in '*Informatics Education: Europe cannot afford to miss the boat*' [EoIE16] and '*Informatics Education in Europe: Are We All In The Same Boat?*' [oECEC17]. Based on the term definitions in section 2.6 nine experts were asked to classify the original competencies from five of the seven selected curricula. The curriculum from England was not part of this study because it provides already the categories '*computer science (CS)*', '*information technology (IT)*', and '*digital literacy (DL)*' [Ber13]. Further, the standards from the German Informatics Society [Ges19] were still under development at the time of the study and could not be categorized. Further, only the standards from the subject *Digital Technology* from the Australian curriculum were part of the survey. Thus, the number of categorized competencies is 22 instead of 28. This survey and its results

were already published by Pasterk and Bollin [PB17a] and Pasterk et al. [PKB19]. Pasterk et al. describe it as follows [PKB19]: 'a group of nine experts, consisting of four computer science teachers and five researchers in the field of computer science education, participated in a survey to categorize the learning outcomes of three selected educational models. To get a larger basis, the survey was repeated with the same group of experts and two additional educational models, the CSTA computer science standards from 2011 and from 2017. Every expert completed a questionnaire including all learning outcomes of the selected models for primary education in a random order and had to choose one of the following categories: *CS*, *DL*, *Both*, or *None*.' The results of this survey are described in section 5.

For a more detailed categorization the method of *coding* was used, like Barendsen et al. [BMD⁺15] or also Hubwieser et al. [HGB⁺15] describe it. As proposed in literature, the process was separated into the three steps *open coding*, *axial coding*, and *selective coding* [CMM18].

Open coding: during the first phase all the original competencies from the different curricula were analyzed and each phrase was assigned to an open code, similar to the phrase itself but more general. For categorization aspects the verbs were not considered. This step lead to 310 *open codes* and 618 coded elements.

Axial coding: after finishing the open codes, the next phase was used to summarize them to more general categories, referring to the same context. During this process 129 *axial codes* were defined.

Selective coding: during the last phase, all of the 129 axial codes were assigned to one of the 27 subcategories of the system proposed by Duncan and Bell [DB15] and described in section 3.2.

These three steps lead to categorized competencies of all seven curricula. A comparison of the distribution of elements within the categories can be found in section 5.5.

4.6.4 Combination to a Generic, Graph-Based Model for Competencies

After prior steps the process of combining the graphs started with a first collection of similar competencies within the categories. In this context, similar means that competencies deal with the same topic on the same cognitive level. The determination of the cognitive level is a challenge on its own, as the formulations are not limited to specific verbs and make use of a lot of different synonyms. For instance, in formulations concerning programming the verbs 'create', 'develop', or 'implement' a program occur. It happens in several cases that very similar activities are described

by different verbs. Additionally, the set of action verbs categorized by Anderson and Krathwohl is also limited. There exist a lot of online resources like additional lists of verbs. But some of them are inconsistent or again very limited. Therefore, synonym lists from online resources like *Woxikon*² or *Wordnet*³ are used to create an own dictionary, based on appearing words in the curricula documents. This was done manually in a first step and was automated in a second step. Further description of this process can be found in section 8.4.

The idea to combine the single curricula to one generic model is based on a construct that is called *intersector nodes*, which is defined as follows:

Definition 4.27 (Intersector node). *Intersector nodes represent a collection of similar competencies which cover the same topic and are located on the same cognitive level.*

The name 'intersector' reflects the main meaning of the nodes, as they represent the intersection of different curricula, based on semantically equivalent competencies. *Intersector nodes* contain the attributes 'ID', 'text', a list of included competencies, a list of countries, a minimum age and a maximum age. As 'text' the most general formulation of the included competencies is taken. In most cases this is the formulation that covers all the others. The minimum/maximum age is determined by the lowest/highest minimum/maximum age of the included competencies. A straight forward example of an *intersector node* is the combination of the following competencies:

1. from the Australian Curriculum [Aus13]

Students are able to identify how common digital systems (hardware and software) are used to meet specific purposes.

2. from the Austrian digital competency model [Dig13b]:

Students are able to cite important application areas of information technology from the environment, they live in.

3. from the model from Berry for the English curriculum [Ber15]

Students are able to discuss common uses of information technology beyond school.

4. from the standards of the German Informatics Society [Ges19]:

² <https://synonyms.woxikon.com/>

³ <https://wordnet.princeton.edu/>

Students are able to explain that computer science is omnipresent in their world.

In this case the topic is information technology or digital systems in the environment of the students. So, the topic is very similar. Slightly different are the verbs and with that the activities. Here, 'discuss', 'identify', 'cite', and 'explain' are used in the four competencies. In Bloom's revised taxonomy from Anderson and Krathwohl (see section 2.5.2) 'cite' and 'identify' are in the cognitive level *remember (level 1)*, but 'discuss' and 'explain' can be found in the level *understand (level 2)* [AK01]. That means, competencies 1 and 2 are combined to an *intersector node* with the text of competency 1, as well as competencies 3 and 4 with the text of competency 4. This is a straight forward example, because it only affected original competencies without subcompetencies, the content is clear and the cognitive levels are easy to distinguish. The next example shows the benefit of the standardization because their formulations are very similar, except for a few minor differences. Before the example can be presented, the question, if one *intersector node* can contain more than one competency or subcompetency from one curriculum, has to be clarified.

If a topic is in one model A covered in a general way by one single competency and in another model B in a detailed way by even more than one competency, then they are still combined to an *intersector node* and all competencies from model B covering this topic are included. That means, an *intersector node* can consist of more than one competency from one curriculum. This is necessary because, although the competencies are in a standardized form, still the granularity of their formulations can be different.

After this clarification the promised example for the combination of two competencies from one curriculum with competencies from other curricula is presented. The list shows the original competencies followed by all of their subcompetencies.

1. from CSTA Standards of 2017 [CST17]:

Students are able to develop programs with sequences and simple loops, to express ideas or address a problem.

- (a) Students are able to develop programs with sequences, to express ideas.
- (b) Students are able to develop programs with sequences, to address a problem.
- (c) **Students are able to develop programs with simple loops, to express ideas.**
- (d) **Students are able to develop programs with simple loops, to address a problem.**

2. from the English national curriculum [Nat14] with subcompetencies from the model from Berry [Ber15]:

Use sequence, selection, and repetition in programs; work with variables and various forms of input and output.

- (a) Students are able to use sequence in programs.
- (b) Students are able to use selection in programs.
- (c) **Students are able to use repetition in programs.**
- (d) Students are able to work with variables.
- (e) Students are able to work with various forms of input.
- (f) Students are able to work with various forms of output.

3. from the curriculum from Switzerland [Leh14]:

Students are able to write and test programs with loops, conditional instructions and parameters.

- (a) **Students are able to write programs with loops.**
- (b) Students are able to write programs with conditional instructions.
- (c) Students are able to write programs with parameters.
- (d) Students are able to test programs with loops.
- (e) Students are able to test programs with conditional instructions.
- (f) Students are able to test programs with parameters.

As the original competencies 1, 2 and 3 of this example show they are very similar each of them contains an element that differs from the others. Competency 1 is limited to 'sequences' and 'simple loops' and additionally the purpose to develop a program is described. The content from 2 is broader adding 'selection', 'variables', 'input', and 'output' to 'sequence' and 'repetition'. Competency 3 does not include 'sequence' but 'loops' and adds 'conditional instructions' and 'parameters'. Thus, all three only overlap in terms of 'loops' or 'repetition'. This applies to subcompetencies (c) and (d) of competency 1, subcompetency (c) of competency 2, and subcompetency (a) of competency 3, which are highlighted in the list. The three verbs used in the subcompetencies are 'develop', 'use', and 'write'. Following Anderson and Krathwohl 'develop' and 'write' are on the level *create (level 6)*, but 'use' is on level *apply (level 3)* [AK01]. But, as 'use' occurs with an 'in a program' it can be compared to 'write a program' and is therefore included in the same *intersector node*. Also both subcompetencies of competency 1 are part of this *intersector*,

because they give more detailed information about the purpose but don't add new specific actions or content. The text for this *intersector node* is 'Students are able to develop programs with loops.'

Another more complex example shows an *intersector node* with more competencies from one curriculum:

1. from CSTA Standards of 2011 [SCF⁺11]:

Students are able to use productivity technology tools for individual and collaborative writing, communication and publishing activities.

- (a) **Students are able to use productivity technology tools for individual writing activities.**
 - (b) Students are able to use productivity technology tools for individual communication activities.
 - (c) Students are able to use productivity technology tools for individual publishing activities.
 - (d) Students are able to use productivity technology tools for collaborative writing activities.
 - (e) Students are able to use productivity technology tools for collaborative communication activities.
 - (f) Students are able to use productivity technology tools for collaborative publishing activities.
2. from the English national curriculum [Nat14] with subcompetencies from the model from Berry [Ber15]:

Use technology purposefully to create, organise, store, manipulate and retrieve digital content.

- (a) Students are able to use technology purposefully to retrieve digital content.
- (b) Students are able to use technology purposefully to store digital content.
- (c) Students are able to use technology purposefully to organise digital content.
- (d) **Students are able to use technology purposefully to create digital content.**
- (e) Students are able to use technology purposefully to manipulate digital content.

3. from the Australian Curriculum [Aus13]:

Students are able to create and organise ideas and information using information systems, and share information in safe online environments.

- (a) **Students are able to create ideas using information systems.**
- (b) **Students are able to create information using information systems.**
- (c) Students are able to organise ideas using information systems.
- (d) Students are able to organise information using information systems.
- (e) Students are able to share ideas and information with known people in safe online environments.

4. from the Austrian digital competency model [Dig13b]:

Students are able to enter texts and format them.

- (a) **Students are able to enter texts.**
- (b) **Students are able to format texts.**

5. from the curriculum from Switzerland [Leh14]:

Students are able to create and present simple picture, text and sound documents.

- (a) **Students are able to create simple picture documents.**
- (b) **Students are able to create simple text documents.**
- (c) **Students are able to create simple sound documents.**
- (d) Students are able to present simple picture documents.
- (e) Students are able to present simple text documents.
- (f) Students are able to present simple sound documents.

This example is complex because of the different granularity of the subcompetencies and the multiple verbs and objects. All have in common that they describe the creation of digital content, but doing this in a more general or more specific way. Competencies 2 and 3 are more general and collect different types of data with phrases like 'create digital content' or 'create information using information systems'. Competency 1 is focusing on writing and communication. Competency 4

is focusing specifically on entering and formatting text and competency 5 collects three different types of documents students should be able to create and present. So, with this the content overlap for subcompetencies (d) of 2 and subcompetencies (a) and (b) of 3, as well as for subcompetencies (a) of 1, (a) and (b) of 4, and (a) of 5 exists. Comparable to the previous example, in subcompetencies (a) and (b) of competency 3 the verb 'create' is used, whereas the verb in (d) of 2 is 'use' with the addition 'to create'. Therefore, they are assumed to be on the same cognitive level *create (level 6)* and combined. In the case of subcompetency (a) of competency 1 again 'use' appears together with 'for writing activities' which can be interpreted as entering and formatting text. This means, (a) of competency 1 and (a) and (b) of 4 are on the same cognitive level *apply (level 3)*. Subcompetencies (a) of competency 5 is obviously related to both groups, but with the verb 'create' not on the same cognitive level as subcompetencies of 1 and 4. With the more specific content 'picture documents', 'text documents', and 'sound documents' the subcompetencies of 5 are too specific to combine them with subcompetencies of 2 and 3. But if all of the three subcompetencies of 5 concerning the creation of documents are included, they become more general and suitable for the mentioned combination. So, subcompetencies (a), (b), and (c) of competency 5 are combined with subcompetencies (d) of 2 and (a) and (b) of 3.

As shown in some cases a combination of more than one competency or subcompetency from one curriculum to one *intersector node* is useful and makes sense. What is not possible is a competency that is part of more than one *intersector node*.

4.7 Conclusion

In this chapter the mathematical basis for a *graph-based approach* is built. Related research shows, that the idea of a graph-based curriculum representation is worthwhile and offers new insights. Based on elements from related approaches and new, self-developed elements, the *graph-based model* displays competencies as nodes containing several attributes. To connect the nodes, two types of relations, *requires* and *expands*, are defined. Together, these components build a *property graph* for competencies, which can be mapped to a *graph database*. The process of the generation of the graphs is based on the identification of comparable elements, the addition of relations, and the evaluation of experts to revise the model. After the representation of all single curricula as graphs, they are combined to a *Generic, Graph-based Model for Competencies*. Therefore, the competency formulations are standardized following a sequence of steps. Further, a categorization is necessary, to collect those competencies, which deal with the same topics. This is done by an *expert rating* as well as *coding activities*. In a last step, similar competencies are substituted by so called *intersector nodes*, representing the linking points of the curricula. The follow-

ing chapters include additional information about the methods used to analyze and compare the single curricula graphs as well as the GGBMC, and present the results.

5. COMPARISON OF DIFFERENT EDUCATIONAL MODELS

5.1 *Introduction*

The previous chapter deals with the basic theoretical background and the explanation of the graph-based approach applied in this dissertation. Different processes in this research led to several results which are presented and discussed in the current chapter. As the existing research to represent curricula as graphs have already been described, the first section in this chapter includes related work for the comparison and analysis of graphs based on curricula. Findings and approaches from this related work contributed to the methods used in this dissertation. The determined metrics are defined and explained in section 5.3. After these clarifications, the results for the metrics in the single curricula graphs are presented in section 5.4. It includes the comparison of the values for the basic metrics, the identification of central competencies, and an analysis of the structure of the graphs. The categorization mentioned in section 4.6.3 is also part of the comparison as it adds interesting information about the focus of curricula and the distribution of the competencies over the categories. Results from the categorization are discussed in section 5.5.

5.2 *Research in Graph Theoretic Approaches*

Approaches to represent curricula as graphs are already discussed in section 4.3. A few further publications also include mathematical calculations to determine more information about the curricula. An example based on graph theory comes from Lightfoot [Lig10]. In his article, Lightfoot focuses on the improvement of the structure of Bachelor degree curricula. Further, he searches for the correct placement of assessment within them. For this purpose the courses of a curriculum are mapped to nodes and the prerequisite requirements display the edges between them. As these relationships have a direction, simple acyclic directed graphs are the results of this process. Nodes that are not connected to other nodes, he calls '*isolated*'. For calculation he uses *measures of degree*, *measures of centrality*, and *clustering measures*. The *degree* of a node represents the number of connected edges (see definitions 4.4 and 4.8). Lightfoot uses both the *in-degree* as well as the *out-degree*. With the help of the *out-degree*, fitting initial courses are determined which offer the possibility of baseline assessment. So, courses with a high value of the *out-degree*

should include 'introductory material or framework concepts' [Lig10]. Nodes with a high value in the *in-degree* indicate high-level courses with appropriate learning activities. They represent courses with a potential for final assessments. With the *centrality measures* important nodes in graphs or networks are identified. There are different types of centrality and Lightfoot focuses on the *betweenness centrality* and the *eigenvektor centrality*. The *betweenness centrality* helps finding courses which build bridges between independent 'course program tracks or course clusters' [Lig10]. Nodes with a high *eigenvektor centrality* are adjacent to nodes with high degree values, and therefore, represent good points for reinforcement before assessments. Early courses with a high *eigenvektor centrality* can also be used to 'introduce topics and perform baseline assessment' [Lig10]. As the measure, the *clustering coefficient* is also well known in network analysis to identify strongly or 'densly' connected nodes. It compares the density of existing edges to the overall possible connections [Lig10]. That means, courses with a high *clustering coefficient* are most suitable to implement changes into the curriculum, when feedback of any kind indicates a problem. Lightfoot concludes that his model delivers a starting point to build a learning scheme, but needs some 'fine tuning and customization' [Lig10].

A further graph-theoretic approach is presented by Marshall [Mar12], aiming at identifying major changes in the three versions of the undergraduate degree Computer Science curricula from *The Joint Task Force on Computing Curricula, Association for Computing Machinery (ACM) and IEEE Computer Society*. In the years 2001 [TJTFoCCS01], 2008 [TJTFoCCS08] and 2013 [JTFoCCS13a] an ACM/IEEE Computer Science Curriculum was published. In a first step, Marshall simply compares the curricula by numbers, like numbers of knowledge areas or core topics, to determine a change of focus. After this first analysis, the curricula are mapped to graphs, using the structure of an onion: in the center stands the discipline 'Computer Science', followed by the three rings containing the knowledge areas, the knowledge units and the topics in this sequence. The knowledge areas, knowledge units and topics represent the nodes, and their relationships the edges of the graphs. As the relationships are directed, the results are directed graphs. A comparison of the *visual representations* is used to get a quick overview about the similarities and differences between the three versions of the curriculum. It is mentioned that this representation does not provide detailed information about similarities and differences, but an idea, where to look at. This more detailed look is implemented in an *algorithmic comparison*. The presented algorithm gets two graphs G_1 and G_2 as input and calculates a graph G_3 using the nodes from G_2 and the structure from G_1 . So, a comparison between G_3 and G_1 indicates, if G_2 is compatible with G_1 . Such a graph is generated for each combination of the three versions of the curriculum. Afterwards, the number of nodes and edges of the combined and the generated graphs are compared to show changes in the *size* of the curricula. To get information of the *overlap* of the curricula, the subtraction of graphs is used. If A and B are graphs,

then $A - B$ results in A without all aspects, nodes and edges of B . This process leads to a percentage of overlapping elements. Results of this analysis show that the content of Computer Science curricula changed from 2001 to 2013 significantly in size and in similar elements [Mar12]. In her dissertation titled 'A graph-based framework for comparing curricula' Marshall explains her approach and describes the mathematical background in detail [Mar14].

The graph-based approach of this contribution uses ideas and methods of Lightfoot [Lig10] and of Marshall [Mar12], focusing on curricula, educational standards, and competency models for computer science in primary education. Elements like comparing the size by the numbers of nodes and edges, and the identification of central elements by centrality measures are part of this approach. In contrast to the two approaches mentioned, in this dissertation two different types of dependency relations, *expands* and *requires*, are used and analyzed. A further difference is the technology used for the necessary calculations which in case of this contribution is a graph database with its own query language. Also, some additional metrics like e.g. *PageRank* are added and some are not considered at all. In the following section the metrics used are described.

5.3 Graph-Based Metrics

5.3.1 Basic Metrics

Some values of the graphs are used as *metrics* to make a comparison possible. In this and the following sections the *metrics* and their interpretation in the context of curricula analysis is explained. Comparable to the approach of Marshall [Mar12], the *size* of the curricula is determined by counting nodes and edges of their graph-representations. Additionally, the *density* of the graphs provides information about the ratio of nodes and edges. The density of a graph is calculated by dividing the number of edges by the total number of possible edges. In a directed graph the number of possible edges is $|V| \times |V| - 1$ with $|V|$ representing the number of nodes. Therefore, the formula for the density of a graph is [Pre12, p. 167]

Definition 5.1 (Density).

$$D = \frac{|E|}{|V|(|V| - 1)}$$

A ratio between nodes and edges provides information about the complexity of graphs and networks. The more edges, the more complex the structure of a graph is [Pre12]. In this approach, the *density* is used to determine the interrelation of the competencies within a curriculum. This can lead to assumptions about the complexity of the curricula. The *degree* is a further value that is compared. Nodes

with a high degree value are related to a lot of other nodes (see definition 4.4) and can be seen as *central nodes*. This application of the *degree* is also known as *degree centrality* [Pre12] and is, besides other centrality measures, applied to identify *central competencies* in curricula. Here, the *in-* and *out-degree* provide additional information, differing between incoming and outgoing relations in directed graphs (see definition 4.8). Competencies with a high *in-degree* are *expanded* or *required* by a lot of other competencies. This indicates that these competencies are necessary for different, independent competencies and can be interpreted as basic and central competencies. A high *out-degree* occurs in competencies which are dependent on many other competencies. These can be assumed to connect different topics or competency paths.

5.3.2 Centrality Measures

There are several possibilities to measure the centrality of nodes within a graph or network. The most basic and intuitive one is the already mentioned *degree centrality*. The formula to calculate *degree centrality* looks like follows [Pre12, p. 97]:

Definition 5.2 (Degree Centrality).

$$C_D(i) = \sum_{j=1}^n x_{ij} = \sum_{i=1}^n x_{ji}$$

Where,

x_{ij} is 0 or 1 depending if a connection between node i and node j exists, and

n is the number of all nodes in the graph.

Comparing the centrality of different graphs the normalized formula should be used [Pre12, p. 97]:

$$C'_D(i) = \frac{C_D(i)}{n - 1}$$

This general *degree centrality* does not take direction into account. For digraphs the *in-degree* and *out-degree centrality* can be distinguished and calculated as follows [Pre12, p. 100]:

Definition 5.3 (In- and Out-Degree Centrality).

$$C_I(i) = \sum_{j=1}^n x_{ji}$$

$$C_O(i) = \sum_{j=1}^n x_{ij}$$

Where,

x_{ij} is 0 or 1 depending if a connection between node i and node j exists, and n is the number of all nodes in the graph.

The normalized formulas look as follows [Pre12, p. 97]:

$$C'_I(i) = \frac{C_I(i)}{n-1}$$

$$C'_O(i) = \frac{C_O(i)}{n-1}$$

In network analysis the *degree centrality* is seen as a limited centrality measure, as it only considers the immediate connections of a node and ignore the rest of the graph [Pre12]. Another possibility to determine central nodes is the *betweenness centrality* which is also applied by Lightfoot [Lig10]. This measure does not focus on the connections of a node but on its placement within the graph. So, for a high *betweenness centrality* value it is important if and how often a node connects disconnected elements of the graph. To determine *betweenness centrality* of a node, the frequency of shortest paths passing through this node is calculated, following this formula [Pre12, p. 105]:

Definition 5.4 (Betweenness Centrality).

$$C_B(k) = \sum_{i \neq j \neq k} \frac{\delta_{ikj}}{\delta_{ij}}$$

Where,

δ_{ikj} is the number of shortest paths between nodes i and j passing through node k .

δ_{ij} is the number of all shortest paths between nodes i and j

The value can be normalized as follows [Pre12, p. 105]:

$$C'_B = \frac{C_B(k)}{\left[\frac{(n-1)(n-2)}{2} \right]}$$

With,

n being the number of all nodes in the graph.

In the context of curricula the *betweenness centrality* is used to identify competencies that connect topics which otherwise would not be connected, and have therefore a central meaning for the curricula.

A very influential method to measure centrality is the so called *PageRank*. It was developed by Larry Page, co-founder of Google, to whom it also owes its name. The algorithm considers transitivity in the centrality calculations which is the major difference to other centrality measures. That means, not the importance of a node on its own but of its neighbors, and their neighbors and so on, is of relevance [NH19]. In the context of curricula it needs a slight modification to show important nodes. As in the original version the algorithm follows the information flow, concerning a dependency relation, the direction against the flow is of interest. Competencies which a lot of important neighbors depend on, are of great interest, as they build the basis for later content. Berberich et al. use the following formula for the *PageRank* calculation [BBWV07]:

Definition 5.5 (PageRank).

$$C_P(v) = (1 - \epsilon) \sum_{(u,v) \in E} \frac{C_P(u)}{\text{out}(u)} + \frac{\epsilon}{|V|}$$

Where,

$\text{out}(u)$ is the out-degree of node u .

ϵ is the probability of making a random jump because of no outgoing connections.

Nodes without incoming edges are assigned with the following value [BBWV07]:

$$C_{Plow} = \frac{\epsilon}{|V|}$$

Following Berberich et al. does not consider nodes without outgoing edges. They therefore propose a different calculation of this score [BBWV07]:

$$C_{Plow} = \frac{1}{|V|} (\epsilon + (1 - \epsilon) \sum_{d \in D} C_P(d))$$

Where,

D is the set of nodes without outgoing edges and $D \subseteq V$.

This is further used to normalize PageRank [BBWV07]:

$$C'_P(v) = \frac{C_P(v)}{C_{Plow}}$$

The mentioned modification is used in the formula instead of the *out-degree* the *in-degree*.

All the calculations for basic metrics and centrality measures are implemented in the developed *GECKO*¹ platform which is described in chapter 7. The results are checked and verified with the free graph editor *yEd*² from yWorks and the Python package *NetworkX*³.

5.3.3 Structure and Categorization

Assumptions about the structure of a curriculum can be made from various points of view. They can be based on graph-theoretic metrics or on the visual analysis of the graph itself (see [Mar12]). In this section, some metrics are discussed to get information about the structure. A first point of interest, also for centrality calculations, is the number of appearing *cycles* within the curricula graphs (see 4.12). *Cycles* can indicate structural problems as competencies transitively depend on each other. Therefore, the number of occurring *cycles* in the graphs is part of the structural analysis of the curricula.

In- and *out-degree* are used in a further way, as Pasterk and Bollin describe it [PB17b]: 'with the help of the out-degree and the in-degree special nodes can be identified, *sources* and *sinks*. *Sources* represent the nodes with no incoming relations, *sinks* are nodes without outgoing relations. In curricula, *sinks* can be interpreted as good competencies to start with because they require no prior knowledge within this subject. *Sources* in curricula refer to nodes that mark the end of an topic and are not continued, or represent the knowledge level that should be reached till the end of a given school grade, and are therefore meant to be continued.' The number of *sinks* and *sources* gives information about the structure, as they indicate the number of started and ended topics. Additionally, nodes with a degree of zero are also of interest, as it is described by Pasterk and Bollin [PB17b]: 'these nodes have no relations to any other node of the graph. This indicates that they cover knowledge that is more or less independent from other knowledge items which can signify structural problems or topics.' Of course, these competencies can also start topics that are planned to continue in later grades.

The number of connected components can also provide some information about the structure of curricula, educational standards, and competency models. In this work the weakly connected components are considered which means that in one component not all nodes must be reachable from every other node. Pasterk and Bollin interpret the number of connected components as follows [PB17b]: 'The higher the number of these components is, the more vertex clusters are not connected to each

¹ <https://gecko.aau.at>

² <https://www.yworks.com/products/yed>

³ <https://networkx.github.io/>

other. This can indicate more independent content areas. It has to be considered that vertices without any connection are also counted as single connected components.’ As already mentioned for centrality measures, a comparison of the number of connected components needs some kind of normalization. In this case the relative frequency is calculated. So, the number of connected components in a curriculum is divided by the number of overall competencies in this curriculum. To improve the readability of the results they are multiplied by 100.

The categorization, described in section 4.6.3 is not only part of the combination process, but adds also information to the analysis and comparison of the curricula. So, the results from experts rating as well as from coding process are presented to make assumptions about the focus of each curriculum [PB17a].

5.4 Results of the Comparison

5.4.1 Methods

The metrics defined in the previous section are applied to two sets of data: these are on the one side the *original competencies* as they are retrieved from the curricula, and on the other side the *standardized competencies* which are prepared following the process described in 4.6.2. So, in the *standardized competencies* so called *competency clusters* (see 2.4.2), which represent competencies that can be divided into *subcompetencies*, are substituted by their *subcompetencies*. Each *competency cluster* contains at least two *subcompetencies*. In the *standardized competencies* the competencies which are no *competency clusters*, stay in their original version. That means the *standardized competencies* are a combination of *original competencies* and *subcompetencies* substituting *competency clusters*. At some points in the following sections a comparison of the *original* and the *standardized competencies* is included.

5.4.2 Basic Metric Value Comparison

One of the most basic metric values, the *number of nodes*, is already described and analyzed in section 3.6. In the current section the number of competencies is compared to the number of nodes after the standardization process. In Tab. 5.1 the number of original competencies, the number of competency clusters, the number of subcompetencies, and the number of standardized competencies is shown. The *competency clusters* are a subset of all competencies, and reflect competencies which are divided into subcompetencies. At first view one can see that the number of original competencies are very close between 39 and 49. But the number of *competency clusters* and *subcompetencies* fluctuate between 8 and 34 respectively between 23 and 121. The model from Berry for the English national curriculum [Ber15] shows with 8 the lowest number of *competency clusters* and with 23 also the fewest *sub-*

Tab. 5.1: Numbers of original competencies and standardized subcompetencies

Number of	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Original number of competencies	49	43	44	42	47	43	39
Number of competency clusters	32	33	33	15	8	28	34
Number of subcompetencies	78	121	105	44	23	88	106
Number of standardized competencies	93	131	116	71	62	103	112

competencies, although its number of original competencies is the second highest. This can be explained by the fact that the model of Berry is based on the original English curriculum and already divides the original statements in a similar manner as the here presented standardization process does. So, only 8 *competency clusters* are left and are divided into 23 *subcompetencies*. With 34 the CSTA standards from 2017 consist of the most *competency clusters* and with 121 the Australian curriculum has the highest number of *subcompetencies*. These values are also reflected by the numbers of *standardized competencies* which are a combination of all original non-cluster competencies and the subcompetencies.

The numbers of subcompetencies per competency cluster are shown in Fig. 5.1. It is visible that, although the numbers of clusters and subcompetencies vary, the numbers of subcompetencies per competency cluster seem to be on a comparable level, except two in special cases. Here, the highest value with 3.67 subcompetencies per cluster appears again in the Australian curriculum, whereas the Austrian competency model has the lowest value with 2.44 subcompetencies per cluster. The numbers of the other five curricula fluctuates between 2.88 and 3.18. This shows that the formulations of the curriculum 21 from Switzerland, the GI standards from Germany, the model from Berry for the English curriculum, and the two CSTA standards from 2011 and 2017 use comparable formulations. Although, the numbers of competency clusters in the Australian curriculum and the Austrian competency model are very close (33 respectively 32, see Tab. 5.1), the difference in the numbers of subcompetencies is very high (121 respectively 78, see Tab. 5.1). So, it can be assumed that the formulations of the competency clusters in the Australian curricu-

lum contain on average one activity or object more than the formulations of the Austrian competency model. Here, the number of clusters does not make all the difference. In the other five curricula the number of competency clusters effect the overall number of subcompetencies.

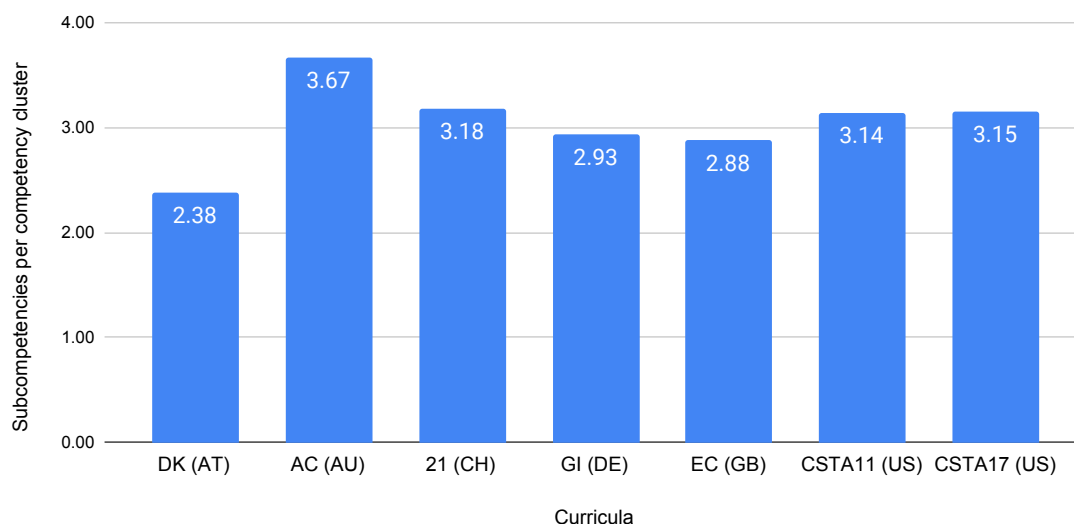


Fig. 5.1: Comparison of the subcompetencies per competency cluster

The numbers of subcompetencies have a direct effect on the standardized curricula models, as competency clusters are substituted by their subcompetencies (see section 4.6.2). A comparison of the absolute numbers of original and standardized competencies can be seen in Fig. 5.2. As already discussed, the Australian curriculum has a very high number of subcompetencies per competency cluster and therefore also contains the most standardized competencies (131 competencies). This is a large increase compared to the number of original competencies. The second and third highest values can be found in the curriculum from Switzerland (116 competencies) and the CSTA standards from 2017 (111 competencies). The curriculum of the GI in Germany with 71 competencies and the model for the English curriculum with 62 competencies contain the lowest number of standardized competencies. This comparison shows that although the numbers of original competencies are not that different, the formulations within the curricula differ in the usage of verbs or objects. It has to be mentioned that these results do not enable assumptions about the granularity of the competency formulations.

The *number of relations* is a further basic metric that is discussed in this section. The number of *expands-* and *requires-relations* are compared for the original curricula as well as for the standardized ones. As seen in Fig. 5.3 the number of

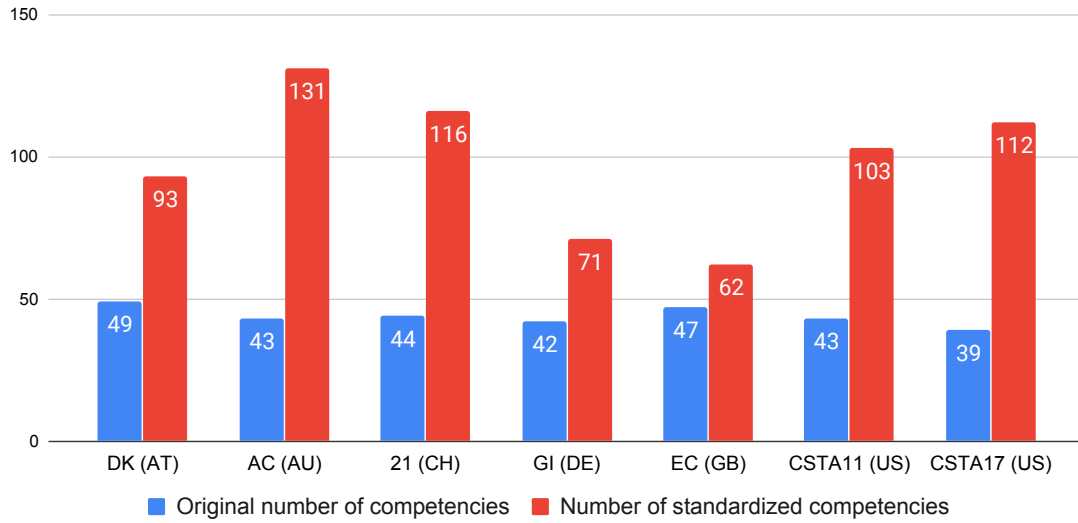


Fig. 5.2: Comparison of the absolute numbers of original competencies and standardized competencies

expands-relations in all curricula is higher than the number of *requires-relations*, except the curriculum from Switzerland with 28 *requires-relations* and 23 *expands-relations*, and the CSTA standards from 2011 with the equal number of 29 *expands-* and *requires-relations*. The model of Berry for the English curriculum contains with 3 the lowest number of *requires-relations* and with 43 the highest number of *expands-relations*. The highest number of *requires-relations* show the CSTA standards from 2011 with 29. With 63 the Austrian competency model contains the most relations overall, whereas the GI standards from Germany contain with 37 the lowest number of overall relations.

As the curricula contain a different number of competencies, the relations per competency represent a possibility of normalization. Fig. 5.4 shows the *expands-* and *requires-relations* per original competency. As the numbers of competencies in the original curricula is very similar, also the results of this normalized values are similar to those presented in Fig. 5.3. Again, the model for the English curriculum contains the most *expands-* (0.91 per competency) and the fewest *requires-relations* (0.06 per competency) per competency. With 0.52 the curriculum from Switzerland presents the lowest number of *expands-relations* per competency. The highest number of *requires-relations* per competency show the CSTA standards from 2011 with 0.67. Overall the CSTA standards from 2011 contain with 1.35 the most relations per competency. The GI standards from Germany show with 0.88 the lowest number of overall relations per competency.

Fig. 5.5 shows the absolute numbers of relations after the standardization pro-

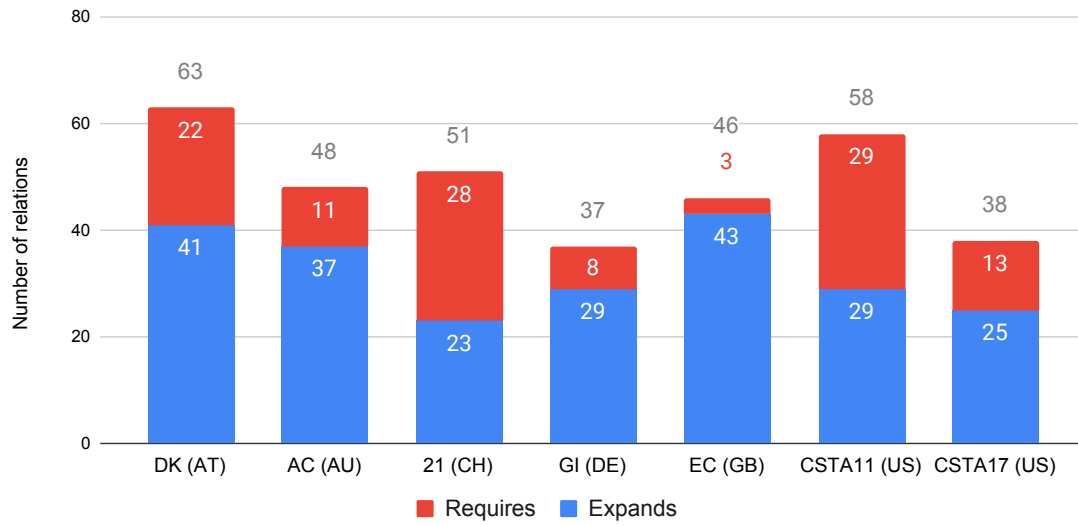


Fig. 5.3: Comparison of the absolute numbers of relations in the original curricula

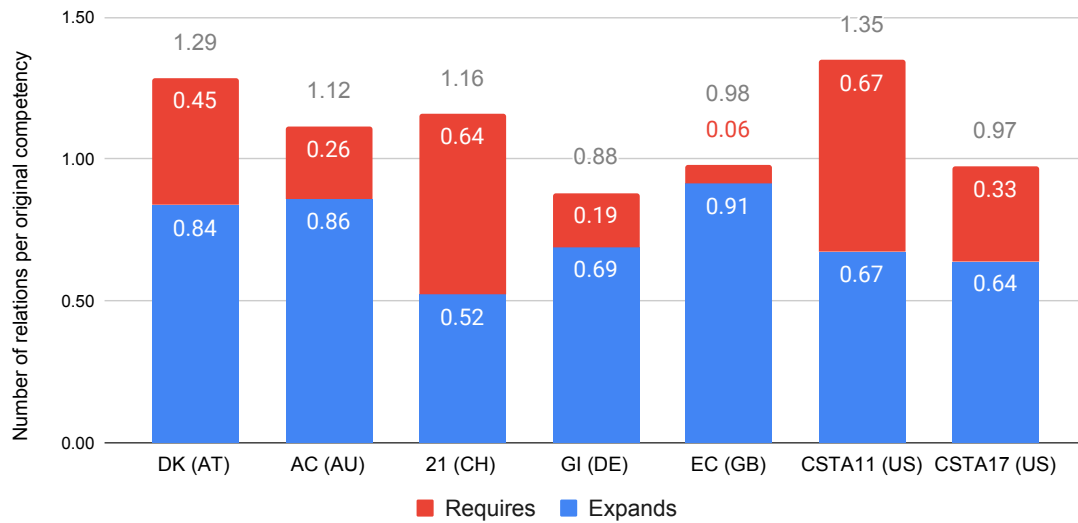


Fig. 5.4: Comparison of the number of relations per original competency

cess. Like it is explained in section 4.6.2, the divided competency formulations are connected by *expands-relations* and only in some cases *requires-relations* are added. At first glance, it is clear that in the standardized curricula the number of *expands-relations* is higher than in the original version. As the numbers of new relations depends on the number of subcompetencies, the Australian curriculum now

contains with 123 the most *expands-relations* (increased by 86) and with 135 the most relations overall. Because of its low number of subcompetencies the model for the English curriculum shows the smallest number of *expands-relations* with 56, of *requires-relations* with 4, and of relations overall with 60. In the original version this curriculum contains the most *expands-relations*. The highest number of *requires-relations* is found in the CSTA standards from 2011, as it is also the case in the original versions. But these standards also show the most during the standardization process added *requires-relations*. Now it contains 34 *requires-relations* (increased by 5). A look at the remaining curricula shows that the Austrian competency model and the GI standards have the same numbers of *requires-relations*, whereas the other curricula have all one *requires-relation* more.

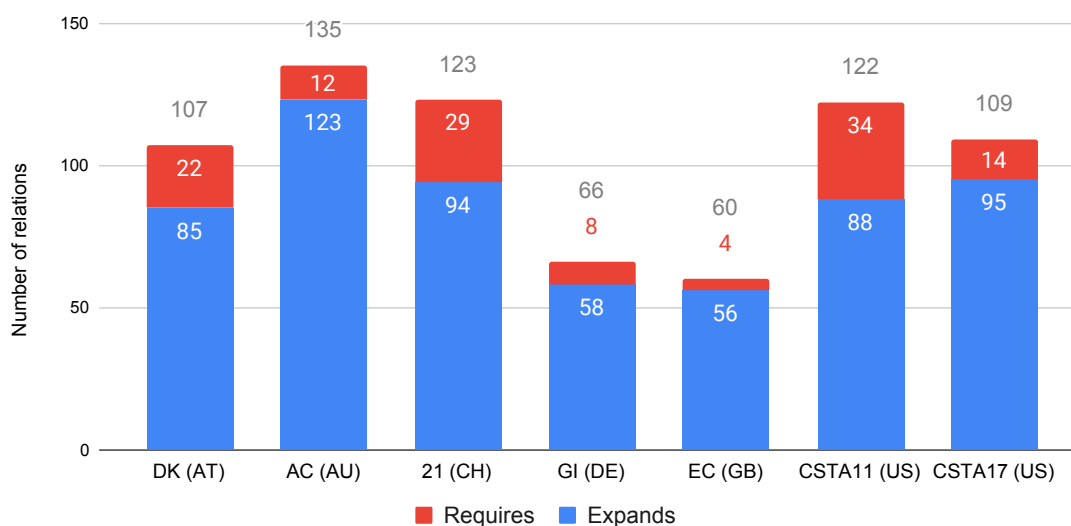


Fig. 5.5: Comparison of the absolute numbers of relations in the standardized curricula

Also, the numbers of relations per competency show, after the standardization, an increase of the *expands-relations*, except in the model for the English curriculum, as it can be seen in Fig. 5.6. Here, a decrease from 0.91 to 0.90 *expands-relations* per competency is visible. The highest increase can be seen in the curriculum from Switzerland which shows 0.29 *expands-relations* more per competency (up to 0.81 from 0.52) in the standardized version. The highest number of *expands-relations* per competency contains the Australian curriculum with 0.94, whereas again the curriculum from Switzerland shows with 0.81 the lowest number. On the other hand, the numbers of *requires-relations* decreased in all curricula, except again in the model for the English curriculum. In the CSTA standards from 2011 the most

requires-relations are found with 0.33 per competency, and the model for the English curriculum still contains with 0.06 the lowest number of *requires-relations* per competency. Overall the CSTA standards from 2011 have the most relations per competency (1.18) and the GI standards the fewest (0.93).

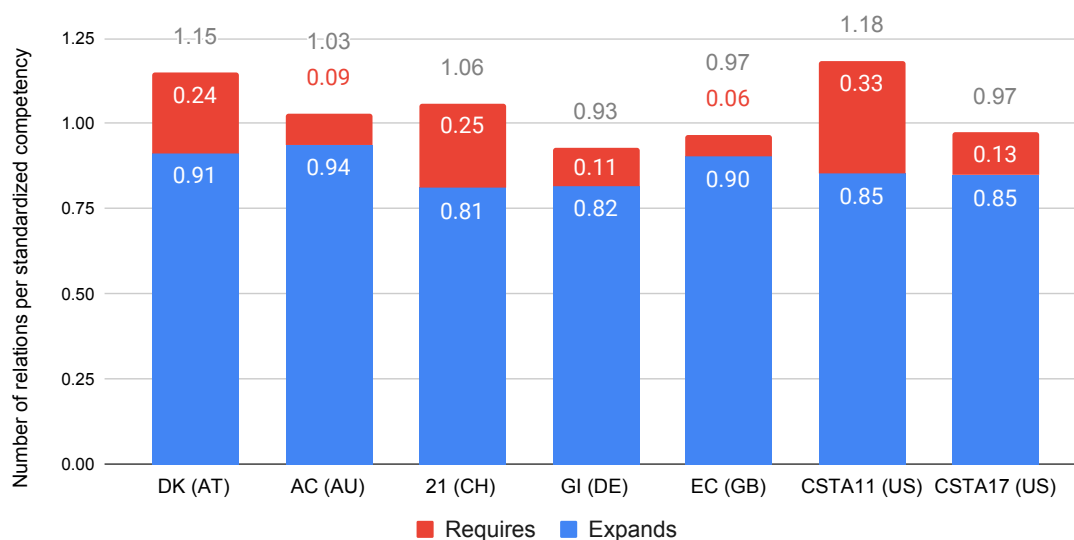


Fig. 5.6: Comparison of the number of relations per standardized competency

The standardization process effects the numbers of relations per competency got closer, as Fig. 5.7 shows. As the overall relations vary in the original version between 0.88 and 1.35 (difference of 0.47), in the standardized version there are between 0.93 and 1.18 (difference of 0.25) relations per competency. This is reflected in the numbers of both relation types. Comparing figures 5.4 and 5.6 the original numbers of *expands-relations* per competency vary between 0.52 and 0.91 (difference of 0.39) and in the standardized version between 0.81 and 0.94 (difference of 0.13). Also, the difference between the *requires-relations* per competency decreased, as they vary in the original version between 0.06 and 0.67 (difference of 0.61) and after standardization between 0.06 and 0.33 (difference of 0.27). This decrease displays an additional and unintended effect of the standardization process.

The last basic metric is the *density*, already defined in section 5.3 and very similar to the simple relations per competency ratio. As this metric directly depends on the number of nodes in a graph, Tab. 5.2 shows the numbers of nodes and the calculated density values for original as well as standardized curricula. For the original curricula the *density* values are between 2.13 and 2.70, except the CSTA standards from 2011 with the very high value of 3.21. Here, a comparison is easy and viable because the curricula have a similar number of competencies. As Fig. 5.8 shows, the three curricula from Austria, Australia, and Switzerland show very similar density values,

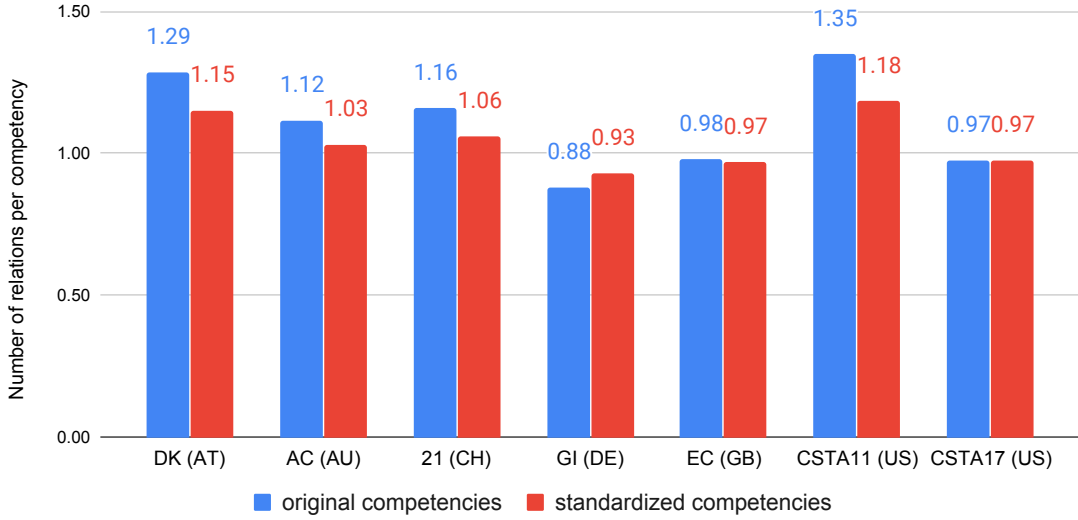


Fig. 5.7: Comparison of the relations per original and per standardized competency

but they differ in the values for *expands-* and *requires-relations*. In the Austrian and Australian curricula the density of *expands-relations* (1.74 respectively 2.05) is higher than the density of *requires-relations* (0.94 respectively 0.61). In contrast, the curriculum from Switzerland has with 1.48 a higher density of *requires-relations* than of *expands-relations* (1.22). Also very close overall density values are found in the CSTA standards from 2017 which has also a higher density of *expands-relations*. The GI standards and the model for the English curriculum have with 2.15 respectively 2.13 the lowest density values. Additionally, the model for the English curriculum shows the lowest density of *requires-relations* and the second highest of *expands-relations*. Outstanding are the CSTA standards from 2011 with the same density values of both relation types and the highest overall density.

After the standardization, the numbers of competencies differ much more which also makes a comparison of the density less meaningful. This is because smaller graphs tend to be more interconnected [Pre12] and therefore show a higher density value. As it can be observed in Tab. 5.2 and Fig. 5.9 this is also the case here. The Australian curriculum shows the highest number of competencies but the lowest density value with 0.79. On the other hand the model for the English curriculum has the fewest competencies but the highest density value with 1.59. Fig. 5.9 shows the density values for the two relations types. It is visible that again the density of *expands-relations* is much higher compared to the original density values.

All metrics described above provide information about the curricula. The *number of competencies* is used to compare the *size* of the models. Additionally, the comparison with the standardized versions highlight differences in the competency

Tab. 5.2: Numbers of original competencies and standardized subcompetencies

	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Original curricula							
Number of competencies	49	43	44	42	47	43	39
Density	2.68	2.66	2.70	2.15	2.13	3.21	2.56
Standardized curricula							
Number of competencies	93	131	116	71	62	103	112
Density	1.25	0.79	0.92	1.33	1.59	1.16	0.88

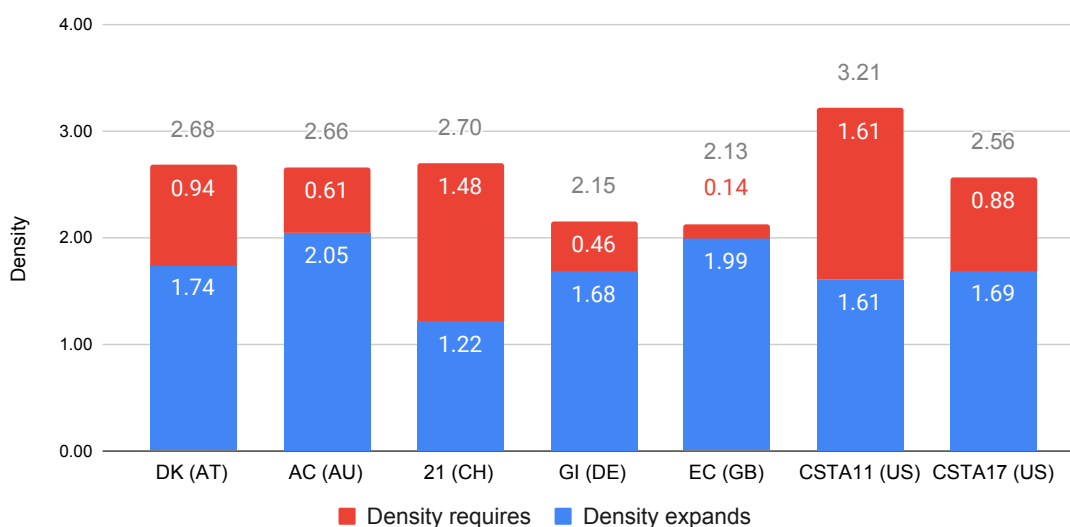


Fig. 5.8: Comparison of the density values for both relation types between original competencies

formulations. Following the results for this metric in the original curricula the Austrian competency model is the most comprehensive curriculum, and the CSTA standards from 2017 the least comprehensive. After the standardization the Australian curriculum contains the highest number of competencies, and the model for the English curriculum has the lowest number. This is not because in the Australian

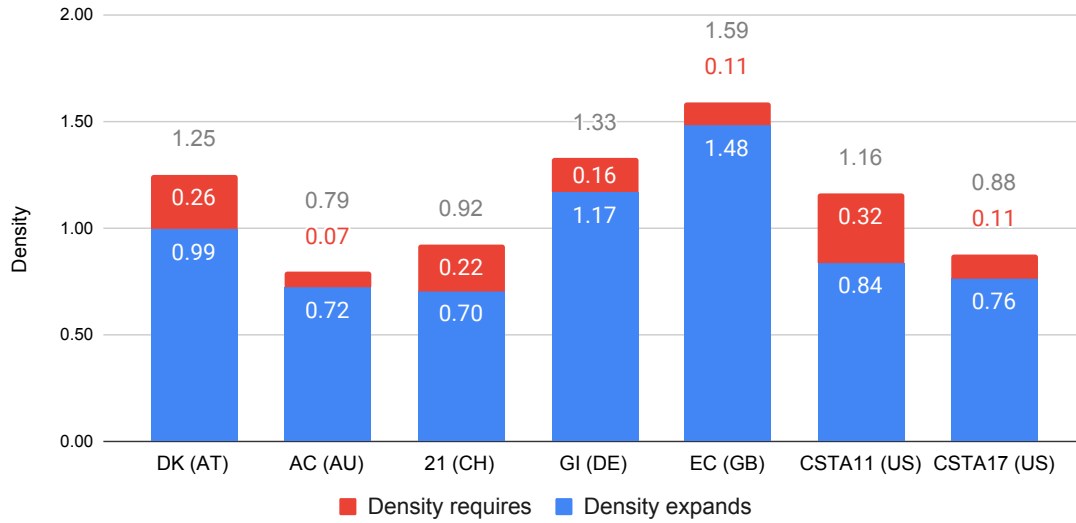


Fig. 5.9: Comparison of the density values for both relation types between standardized competencies

curriculum a lot more competencies, which could be divided into subcompetencies, are found, but because each of the competency clusters included on average one subcompetency more than the clusters in other curricula.

The three metrics *number of relations*, *relations per competency* and *density* can add information about the complexity of curricula, as Pasterk and Bollin already described it [PB17b]. In the original curricula the Austrian competency model and the CSTA standards from 2011 contain the highest number of relations and also the highest number of relations per competency. Considering also the density, where also the CSTA standards have the highest value, it can be assumed that it is the most complex model from the selected ones. The GI standards from Germany include the fewest relations, and show the lowest relations per competency as well as the second lowest density value. Therefore, it seems to be the least complex curriculum in the original version. As mentioned, the standardized curricula are harder to compare as their number of competencies vary much more than in the original ones. In the standardized curricula the Australian curriculum contains the most relations but because of its high number of competencies, not as many relations per competency as the CSTA standards from 2011 or the Austrian competency model. And considering the density, the model for the English curriculum and the GI standards which both have the least number of relations and relations per competency, show the highest values. So, in case of the standardized curricula it is hard to draw any assumption.

Tab. 5.3: Maximum values of degree (including in- and out-degree) in the original and standardized curricula

	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Original curricula							
max. degree	7	4	7	4	4	11	6
max. out-degree	4	3	3	2	2	3	4
max. in-degree	5	3	6	3	3	10	3
Standardized curricula							
max. degree	6	4	7	4	4	14	5
max. out-degree	4	3	3	2	2	3	4
max. in-degree	5	3	6	3	3	13	4

5.4.3 Central Competencies

The identification of special nodes is on the one hand based on simple *degree values* and on the other hand on different *centrality measures*. The first centrality measure used here is the *degree centrality*. It is based on the *degree* of the competencies and gives information about two different kind of central nodes:

- **high in-degree:** basic central competencies
- **high out-degree:** bridge competencies which connect competency paths

As an overview, the highest values of *degree*, *out-degree*, and *in-degree* are presented in Tab. 5.3. On a first view it is visible that the values of the original and the standardized curricula are very similar and differ only in a few cases. In the original and in the standardized curricula the CSTA standards from 2011 show a very high *in-degree value* of 10 respectively 13. This is close to the double of the next highest value and means that there is at least one competency 10 others depend on. The next highest values of *in-degree* appear in the curriculum from Switzerland with 6 and the Austrian competency model with 5. The maximum value of *out-degree* is in the original and the standardized version 4 and can be found in the Austrian competency model and the CSTA standards from 2017.

The following competencies represent those nodes which have the highest *out-degree*, *in-degree*, and *degree* values in the original curricula. It has to be mentioned

that competencies which are included in the maximum values of either *out-* or *in-degree* are not mentioned again under the overall *degree*. For orientation, the country codes and the internal identification numbers of the competencies within the curricula precede the competencies. Here, the formulations from the model for the English curriculum already use a notation which is applied later on for the standardized formulations like e.g. 7.1. This notation represents a subcompetency of the competency with the internal identification number of 7. The number in front of the point states the internal identification number of the competency, the number after the point shows the position of the subcompetency in the sequence of subcompetencies from the corresponding competency. The English model is a special case, as it is based on the teaching intentions from the original curriculum. Therefore, the number in front of the point refers to the corresponding original teaching intention instead of a competency.

maximum out-degree:

- DK/AT [Dig13b]:
 - **AT-26: The students are able to use networks to search and display information.**
- AC/AU [Aus13]:
 - AU-18: The students are able to collect and manipulate different data when creating information and digital solutions.
 - AU-31: The students are able to define problems in terms of data and functional requirements and design solutions by developing algorithms to address the problems.
 - AU-32: The students are able to incorporate decision-making, repetition and user interface design into their designs and implement their digital solutions, including a visual program.
- 21/CH [Leh14]:
 - CH-5: The students are able to learn using predefined media and obtain information on a specific topic.
 - CH-20: The students are able to use media to create and present their work.
 - CH-38: The students are able to distinguish between operating system and application software.
- GI/DE [Ges19]:

- DE-17: The students are able to explain that computer science is omnipresent in their world.
- DE-31: The students are able to control automata by programming.
- DE-32: The students are able to explain the need for a formal language for interaction with computer systems.
- DE-34: The students are able to save and retrieve data.
- EC/GB [Ber15]:
 - GB-7.1: The students are able to design programs that accomplish specific goals.
 - GB-7.3: The students are able to debug programs that accomplish specific goals.
 - GB-12.1: The students are able to design a range of programs that accomplish given goals.
 - GB-12.3: The students are able to design a range of systems that accomplish given goals.
 - GB-12.4: The students are able to create a range of systems that accomplish given goals.
 - GB-12.5: The students are able to design a range of content that accomplish given goals.
- CSTA11/US [SCF⁺11]:
 - US1-15: The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software.
 - US1-17: The students are able to use productivity technology tools for individual and collaborative writing, communication and publishing activities.
 - US1-25: The students are able to understand the connections between computer science and other fields.
 - US1-31: The students are able to use computing devices to access remote information, communicate with others in support of direct and independent learning and pursue personal interests.
 - US1-38: The students are able to identify factors that distinguish humans from machines.
- CSTA17/US [CST17]:

- **US2-10:** The students are able to develop programs with sequences and simple loops, to express ideas or address a problem.

maximum in-degree:

- DK/AT [Dig13b]:
 - AT-17: The students are able to name digital devices of daily life and use them responsibly.
- AC/AU [Aus13]:
 - AU-1: The students are able to identify how common digital systems (hardware and software) are used to meet specific purposes.
 - AU-8: The students are able to identify needs, opportunities or problems and describe them.
 - AU-13: The students are able to safely create solutions and communicate ideas and information face-to-face and online.
 - AU-17: The students are able to explain how the solutions meet their purposes.
- 21/CH [Leh14]:
 - CH-14: The students are able to work with basic elements of the user interface (windows, menu, several opened programs).
- GI/DE [Ges19]:
 - DE-9: The students are able to describe automata in their environment as self-acting machines.
 - DE-14: The students are able to describe that computer systems are designed by humans.
- EC/GB [Ber15]:
 - GB-3.2: The students are able to use logical reasoning to predict what will happen when I read through computer code.
- CSTA11/US [SCF⁺11]:
 - **US1-3:** The students are able to use technology resources (e.g. puzzles, logical thinking programs) to solve age-appropriate problems.

- CSTA17/US [CST17]:
 - US2-1: The students are able to select and operate appropriate software to perform a variety of tasks, and recognize that users have different needs and preferences for the technology they use.
 - US2-8: The students are able to model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.

maximum degree:

- AC/AU [Aus13]:
 - AU-9: The students are able to collect, sort and display familiar data from a range of sources and recognise patterns in data.

The competencies with the highest values of *out-degree* and *in-degree centrality* are highlighted in this list. One competency in the Australian curriculum has a high overall degree (AU-9), but does not appear in the maximum lists of out- and in-degree. This happens because it has as many incoming as outgoing relations. Some of these formulations are very general which is one explanation for the high dependencies. One of these cases is the competency from the CSTA standards with the very high in-degree of 10 (US1-3). As it can be seen it describes that the students should be able to use technology to solve problems which are appropriate for their age. For sure this competency is useful in a lot of situations and it is obvious that several competencies depend on it. The example competencies show that the competencies with high values in *out-degree centrality* are of high level and complexity. Most cases combine several topics or have requirements from different competency paths. Competencies with a high value in *in-degree centrality* represent more basic knowledge and skills. It is clear that they are required or at least useful in a lot of different situations.

During the standardization process some of the original competencies are divided into subcompetencies. This also effects some of the identified central points. In most cases, the *out-degree* of a competency cluster is transferred to the first subcompetency of this cluster and the *in-degree* to the last subcompetency. The following competencies are those with the maximum in-degree and out-degree from the standardized curricula which are not included in the list from the original curricula presented above. Again, for orientation the country code, the internal identification number and the position of the subcompetencies in the sequence precede the competencies. So, for instance the number *AU-26.1* in the Austrian competency model means the first subcompetency of competency 26 in this curriculum.

maximum out-degree:

- DK/AT [Dig13b]:
 - **AT-26.1: The students are able to use networks to search information.**
- AC/AU [Aus13]:
 - AU-18.1: The students are able to collect different data when creating information and digital solutions.
 - AU-31.1: The students are able to define problems in terms of data.
 - AU-32.1: The students are able to incorporate decision-making into their designs.
- 21/CH [Leh14]:
 - CH-5.1: The students are able to learn using predefined media.
 - CH-20.1: The students are able to use media to create their work.
- GI/DE [Ges19]:
 - DE-34.1: The students are able to save data.
- EC/GB [Ber15]:
 - GB-12.7: The students are able to collect data.
 - GB-12.13: The students are able to evaluate information.
 - GB-12.15: The students are able to select a variety of software (including internet services) on a range of digital devices.
- CSTA11/US [SCF⁺11]:
 - US1-2.4: The students are able to work cooperatively and collaboratively with teachers using technology.
 - US1-15.1: The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems.
 - US1-17.1: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual writing activities.
- CSTA17/US [CST17]:
 - **US2-10.1: The students are able to develop programs with sequences, to express ideas.**

maximum in-degree:

- DK/AT [Dig13b]:
 - AT-9.2: The students know how they should behave in a given case concerning the risks associated with the use of information technologies.
 - AT-17.2: The students are able to use digital devices of daily life responsibly.
- AC/AU [Aus13]:
 - AU-8.6: The students are able to describe identified problems.
- CSTA17/US [CST17]:
 - US2-1.2: The students are able to select appropriate software to perform a variety of tasks.

It can be seen that the competencies with the highest *out-degree* values are the first subcompetencies in a competency cluster, whereas the sequence numbers of the subcompetencies with the highest *in-degree* differ as the competency cluster have a different amount of subcompetencies. Of four competencies identified as special in the original curricula no subcompetencies are found in the standardized versions (US1-31, AU-9 and AU-13, US2-8). In the CSTA standards from 2011 and the Austrian competency model special nodes are identified which are not part of competency clusters with the highest degree values in the original curricula (US1-2.4, AT-9.2). The assumptions from the original curricula also apply here, as the competencies with high *out-degree centrality* are of higher level with many requirements and the nodes with high *in-degree centrality* represent basic competencies which are needed in several situations. That means there are only slight differences in the identification of special competencies in the original and the standardized curricula using the *degree centrality*. As the values are very similar the comparison of the normalized values for the *degree centrality* are only discussed for the standardized curricula. Fig. 5.10 shows the differences between the highest values of *degree*, *out-degree* and *in-degree* in the curricula. The highest degree values are higher in all curricula than those of *in-* and *out-degree*. That means that the identified competencies have at least one incoming- and one outgoing relation. In most cases the normalized values of the *in-degree* are higher than of the *out-degree*, except the Australian curriculum and the CSTA standards from 2017 where both values are the same. The Australian curriculum shows the lowest maximum values in *in-* and *out-degree*, the CSTA standards from 2011 the highest maximum values in *in-degree*, and the Austrian competency model the highest *out-degree*. The maximum values differ

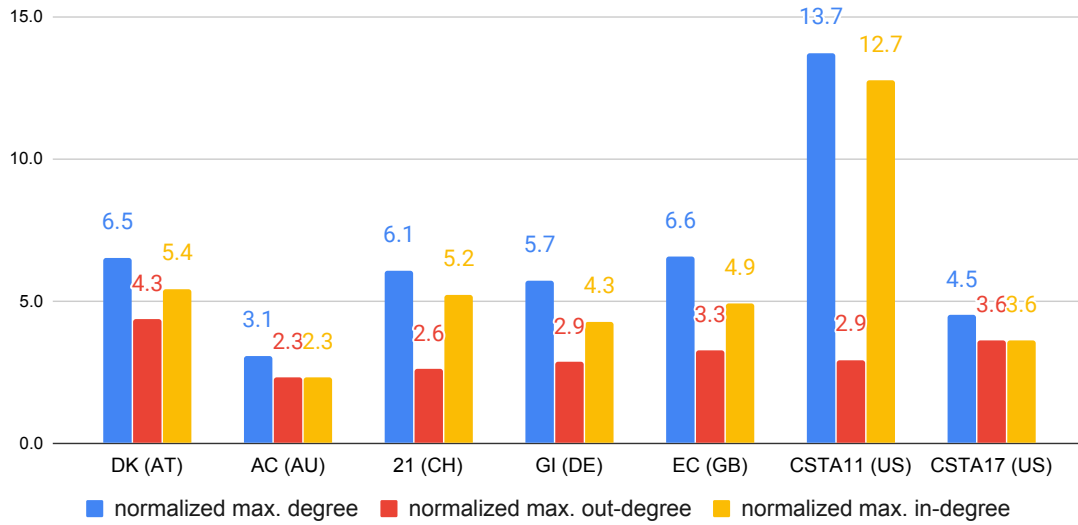


Fig. 5.10: Comparison of the degree values in the standardized curricula

the most in the CSTA standards from 2011, as the *in-degree* values are extraordinary high, followed by the curriculum from Switzerland.

Unlike the *degree centrality*, the measured values of *betweenness centrality* as well as *PageRank* itself are not meaningful on their own. The outcomes which are the as important identified competencies, are of greater interest. Therefore, as results for this measures, the found competencies are presented, without discussing the background numbers. Further, the discussion is limited to the standardized curricula, as the results are similar to those of the original ones, and the subsequent developments are based on the standardized versions. For these two centrality measures the structure of the graphs also has to be considered, as both of them only work for connected graphs. So, if a graph has more than one connected component, the identified nodes are only central for the particular component they belong to. The structures of the graphs representing the curricula are discussed in section 5.4.4.

As mentioned in section 5.3, the *betweenness centrality* shows nodes which connect components. In the context of this dissertation the components are competency or learning paths. Therefore, the nodes are of high interest and handled as central competencies. The following competencies have the highest values for *betweenness centrality* in their curricula:

- DK/AT [Dig13b]:
 - AT-9.2: The students know how they should behave in a given case concerning the risks associated with the use of information technologies.

- AC/AU [Aus13]:
 - AU-18.1: The students are able to collect different data when creating information and digital solutions.
- 21/CH [Leh14]:
 - CH-20.1: The students are able to use media to create their work.
- GI/DE [Ges19]:
 - DE-25.7: The students are able to implement algorithms with the basic algorithmic block repetition.
- EC/GB [Ber15]:
 - GB-11.3: The students are able to explain how to be discerning in evaluating digital content.
 - GB-11.4: The students are able to explain how to use search technologies effectively.
- CSTA11/US [SCF⁺11]:
 - US1-14.4: The students are able to use standard output devices to successfully operate computer related technologies.
- CSTA17/US [CST17]:
 - US2-10.1: The students are able to develop programs with sequences, to express ideas.

It can be seen that the competencies are from different categories and treat different topics. A very strong connecting character show the competencies from the Austrian competency model, the curriculum from Switzerland, and the CSTA standards from 2011. The result for the CSTA standards from 2017 is also comprehensible, since it generally has a strong focus on programming (shown in section 5.5). In the case of the Australian curriculum the identified competency is one of the two competencies with the highest out-degree value. Therefore, it connects several learning paths. The competency found in the GI standards is located in the middle of a long path that is again connected with another long path. That is the reason, why it is identified as connecting node. Concerning the model for the curriculum in England the structure shows, why these two competencies are identified to be central. Here, two major, not connected components can be found: one that includes the competencies for program creation and the other one that handles data management and digital

content creation. For the latter, competencies *GB-11.3* and *GB-11.4* show the highest *betweenness centrality* value.

The *PageRank* values support the search for competencies which are important at the beginning of instructions. Competencies with a high *PageRank* value are basic competencies which a lot of other important competencies depend on. In the following list the competencies with the highest *PageRank* in each curriculum are presented:

- DK/AT [Dig13b]:
 - AT-9.2: The students know how they should behave in a given case concerning the risks associated with the use of information technologies.
- AC/AU [Aus13]:
 - AU-1.0: The students are able to identify how common digital systems (hardware and software) are used to meet specific purposes.
- 21/CH [Leh14]:
 - CH-11.6: The students are able to use simple functions of programs.
- GI/DE [Ges19]:
 - DE-13.0: The students are able to name the components of computer systems using the technical language of computer science.
- EC/GB [Ber15]:
 - GB-3.2: The students are able to use logical reasoning to predict what will happen when I read through computer code.
- CSTA11/US [SCF⁺11]:
 - US1-14.4: The students are able to use standard output devices to successfully operate computer related technologies.
- CSTA17/US [CST17]:
 - US2-8.2: The students are able to model daily processes by creating algorithms (sets of step-by-step instructions) to complete tasks.

All of these competencies are on a very basic level and four of them are important for several topics (AU-1.0, CH-11.6, DE-13.0, US1-14.4). Two of them are also identified as sinks (AU-1.0, DE-13.0). The foci of the curricula presented in section

5.5, provide some insights into the results from *PageRank*. As the Austrian competency model focuses on *human factors and ethics*, also the most important basic competency is from this category. Because of the orientation towards programming, competencies from this category are also identified as central for the model for the English curriculum and the CSTA standards from 2017.

It has to be mentioned that due to limitations of space only those competencies with the highest values are presented. Tab. 5.4 shows the five competencies of each curriculum with the highest values in the discussed centrality measures. For readability only the identification numbers of the competencies within the curricula are included. That means, a number can appear in different curricula and represents different competencies depending of the curriculum. If a number of a competency occurs in one curriculum in several centrality measures they are marked by colors. The values without brackets are the highest values in this measure, those with brackets are the next highest.

In the Austrian competency model it can be seen that the competency with the highest value in *in-degree*, *betweenness centrality*, and *PageRank* is the same. Furthermore, in three cases *betweenness centrality* and *PageRank* identify the same competencies as central. In the Australian curriculum the *out-degree centrality* and the *betweenness centrality* share the competency with the highest value, as do the *in-degree centrality* and *PageRank* with the two top competencies. A similar situation can be seen in the curriculum from Switzerland, where *out-degree centrality* and *betweenness centrality* again share two of the five top competencies. *In-degree centrality* and *PageRank* also share one of the five competencies, as do *betweenness centrality* and *PageRank*. With only two repeatedly identified competencies, the GI standards show the most differences in this analysis. These two are shared by *in-degree centrality* and *PageRank*. In the model for the English curriculum again two competencies with high values in *out-degree centrality* are found under the top five of the *betweenness centrality*, and the competency with the highest value in the *in-degree centrality* also shows the highest value in *PageRank*. One competency from the CSTA standards from 2011 appears in the top five of *in-degree centrality*, *betweenness centrality*, and *PageRank*. *Betweenness centrality* shares two additional competencies with *in-degree centrality* and also with *PageRank*. In the CSTA standards from 2017 the competency with the highest value in the *out-degree centrality* shows also the highest value in the *betweenness centrality*. The *PageRank* shares two competencies from the top five with *in-degree centrality* and one with *betweenness centrality*. So, in five of the seven curricula some kind of relation between *in-degree centrality* and *PageRank* can be assumed. Four curricula show exemplary overlaps between *out-degree centrality* and *betweenness centrality*, three overlaps between *betweenness centrality* and *PageRank*, and again three overlaps between *in-degree centrality* and *betweenness centrality*.

Tab. 5.4: Comparison of the identified competencies by different centrality measures (the colors highlight competencies reappearing in different measures)

	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
max. out-degree	26.1	18.1	5.1	17.0	7.1	2.4	10.1
	(43.0)	31.1	20.1	31.0	7.3	15.1	(28.1)
	(19.1)	32.1	38.0	32.0	12.7	17.1	(30.1)
	(5.00)	(32.3)	(7.1)	34.1	12.13	25.0	(7.2)
	(41.0)	(9.1)	(8.0)	38.2	12.14	38.0	(14.4)
max. in-degree	9.2	1.0	14.0	9.0	3.2	3.0	1.2
	17.2	8.6	(27.0)	14.0	(4.2)	(8.0)	(8.2)
	(1.0)	17.0	(5.2)	(15.0)	(6.4)	(14.4)	(4.4)
	(24.4)	(5.2)	(36.2)	(19.1)	(2.2)	(1.6)	(7.1)
	(4.0)	(6.2)	(2.2)	(1.0)	(4.5)	(5.2)	(11.0)
betweenness centrality	9.2	18.1	20.1	25.7	11.3	14.4	10.1
	(9.1)	(18.2)	(20.2)	(25.6)	11.4	(1.3)	(10.2)
	(8.3)	(23.1)	(21.0)	(25.8)	(12.15)	(3.0)	(10.3)
	(8.2)	(23.2)	(5.2)	(25.5)	(7.3)	(14.3)	(10.4)
	(8.1)	(31.1)	(5.1)	(25.9)	(12.7)	(8.0)	(6.2)
PageRank	9.2	1.0	11.6	13.0	3.2	14.4	8.2
	(9.1)	(8.6)	(14.0)	(14.0)	(3.1)	(14.3)	(8.1)
	(1.0)	(8.5)	(11.5)	(6.0)	(1.3)	(14.2)	(1.2)
	(8.3)	(11.0)	(11.4)	(9.0)	(1.2)	(14.1)	(6.2)
	(8.2)	(13.1)	(11.3)	(7.0)	(1.1)	(1.3)	(6.1)

5.4.4 Analysis of Graph-Structure

The number of *cycles* in the graphs of the curricula is an essential information because some centrality measures like *PageRank* only work for acyclic graphs. In our case, none of the seven graphs representing the curricula contains any *cycle*. So, the centrality measures can be applied and no assumptions about structural problems can be made because of existing *cycles*. With the *degree values* central competencies can be found, as well as start- or end-points of topics marked by *sources* and *sinks*. Additionally, competencies without any connection to other competencies, so called *isolated nodes*, can be found, as they have a degree of zero. All these special nodes provide information about the structure of the curricula. A high number of *sinks* indicates that a curriculum starts with a lot of different topics and covers independent areas. From a big number of *sources* it can be assumed that several topics end here or should be continued in later grades. *Isolated nodes* represent competencies without connections to other competencies and can be used to provide students insights into a topic which is not examined further or also continued in higher grades. The numbers of the identified special nodes are presented in Tab. 5.5. As the reading direction of the relations shows the dependency, the *sources* represent nodes which no other nodes depend on, and *sinks* are nodes which do not depend on others. So, *sources* conclude and *sinks* start topics. The results for the original curricula show that the numbers of *sources* are higher in all curricula, than the number of *sinks*. The numbers of *sinks* vary between 4 and 7, the numbers of *sources* between 10 and 21. It can be seen that the CSTA standards from 2011 contain the fewest *sinks* (4) and by far the most *sources* (21). The model for the English curriculum shows with 10 the lowest number of *sources*, and the standards from the GI and the CSTA of 2017 have with 7 the most *sinks*. The number of isolated nodes is highest in the 2017 CSTA standards at 3 and lowest in the Australian curriculum at 0. In the standardized curricula the number of *sources* is again higher than the number of *sinks*. Compared to the original curricula the numbers of *sources* increased in all curricula. Also the number of *sinks* increased in five from the seven curricula, in the model for the English curriculum and the CSTA standards from 2011 they stayed the same. Again, the CSTA standards from 2011 show the most *sources* (29) and the fewest *sinks* (4). The highest number of *sinks* are found in the CSTA standards from 2017 (11) and the lowest number of *sources* occurs in the model for the English curriculum (11). The standardization has the additional effect that the number of isolated nodes is reduced. Only in three curricula (Switzerland, England, CSTA 2011) nodes with a degree of zero are found. This is because also the original isolated competencies are divided into subcompetencies and connected to each other. Therefore, they have connections and a degree value which is higher than zero.

Fig. 5.11 compares the numbers of *sinks*, *sources*, and *isolated nodes* relative to the numbers of competencies of the original curricula. It is visible that the numbers

Tab. 5.5: Numbers of sources, sinks and isolated nodes in the original and standardized curricula

Number of	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Original curricula							
sinks	6	5	6	7	6	4	7
sources	13	11	11	11	10	21	13
isolated nodes	1	0	1	2	1	1	3
Standardized curricula							
sinks	8	6	7	9	6	4	11
sources	16	13	14	14	11	29	19
isolated nodes	0	0	1	0	1	1	1

of *sources* vary between 21.28% and 33.33%, except in the CSTA standards from 2011 with the highest relative number of 48.84%. This means, in this model nearly every second competency is a *source*. In case of *sinks* the highest value of 17.95% can be found in the CSTA standards from 2017 and the lowest of 9.3% in the CSTA standards from 2011. With 7.69% of *isolated nodes* the CSTA standards show the highest value in this category, whereas the Australian curriculum has 0%.

The following examples show *sinks*, *sources*, and *isolated nodes* from the original curricula:

- **sinks:**

- AT-1: The students are able to cite important application areas of information technology from the environment, they live in ([Dig13b]).
- AU-2: The students are able to use digital systems to represent simple patterns in data in different ways ([Aus13]).
- CH-2: The students are able to understand simple contributions in different media languages and are able to talk about them ([Leh14]).
- DE-13: The students are able to name the components of computer systems using the technical language of computer science ([Ges19]).
- GB-4.2: The students are able to use technology purposefully to store digital content ([Ber15]).

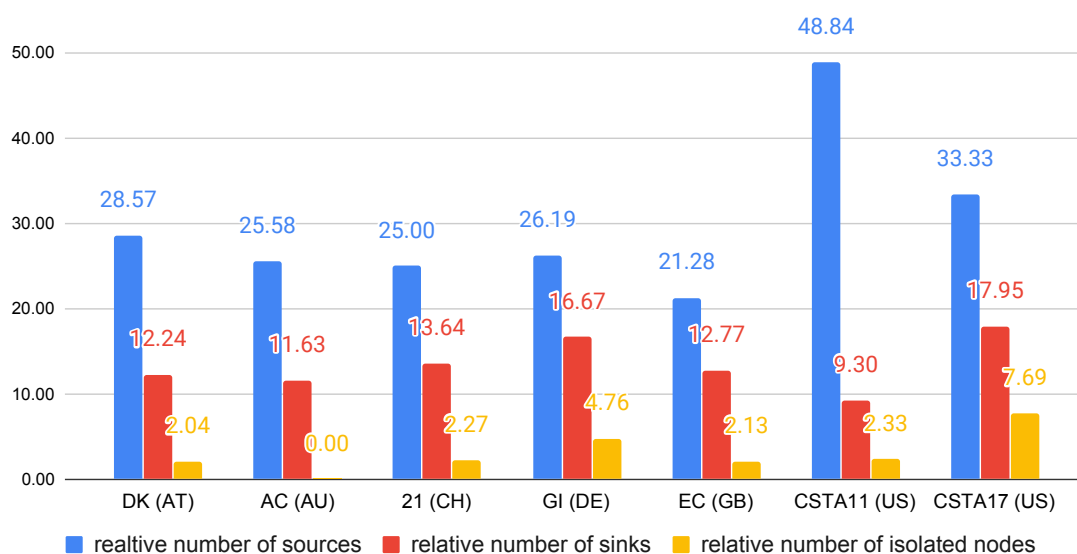


Fig. 5.11: Comparison of the relative number of sources, sinks and isolated nodes in the original curricula

- US1-5: The students are able to understand how to arrange (sort) information in useful order, such as sorting students by birth date, without using a computer ([SCF⁺11]).
- US2-9: The students are able to model the way programs store and manipulate data by using numbers or other symbols to represent information ([CST17]).

• **sources:**

- AT-41: The students are able to use information from the Internet in their work ([Dig13b]).
- AU-30: The students are able to explain how digital systems use whole numbers as a basis for representing a variety of data types ([Aus13]).
- CH-38: The students are able to distinguish between operating system and application software ([Leh14]).
- DE-32: The students are able to explain the need for a formal language for interaction with computer systems ([Ges19]).
- GB-12.3: The students are able to collect, analyse, evaluate and present information ([Ber15]).
- US1-43: The students are able to understand ethical issues that relate to computers and networks ([SCF⁺11]).

- US2-28: The students are able to create programs that include sequences, events, loops, and conditionals ([CST17]).

- **isolated nodes:**

- AT-45: The students are able to encrypt and decrypt some information from everyday life ([Dig13b]).
- CH-30: The students know the names of the document types they use ([Leh14]).
- DE-24: The students are able to use and develop agreements for the transmission of messages ([Ges19]).
- GB-5: The students are able to discuss common uses of information technology beyond school ([Ber15]).
- US1-37: The students are able to identify that information is coming to the computer from many sources over a network ([SCF⁺11]).
- US2-22: The students are able to model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination ([CST17]).

As it can be seen in Fig. 5.12, the relative numbers of *sinks*, *sources*, and *isolated nodes* of the standardized curricula differ from those of the original versions. Apart from the CSTA standards from 2011 which have still by far the highest percentage of *sources*, the relative numbers of *sources* vary between 9.92% and 19.72%. The lowest percentage of *sinks* show again the CSTA standards from 2011, whereas the highest can be found in the GI standards. As there are only four curricula with one *isolated node* each left, the curriculum with the lowest number of competencies, the model for the English curriculum, has the highest percentage.

The following competencies represent examples of *sinks*, *sources*, and *isolated nodes* in the standardized curricula:

- **sinks:**

- AT-20.1: The students are able to start a computer ([Dig13b]).
- AU-8.5: The students are able to identify problems ([Aus13]).
- CH-11.1: The students are able to switch devices on ([Leh14]).
- DE-4.1: The students are able to state that agreements are necessary to encode data ([Ges19]).
- GB-7.6: The students are able to solve problems by decomposing them into smaller parts ([Ber15]).

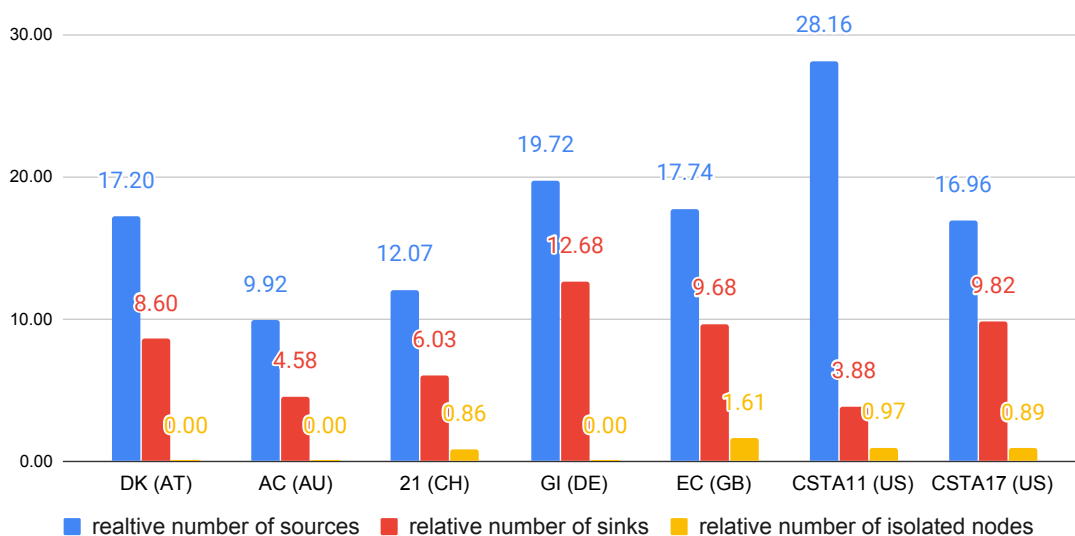


Fig. 5.12: Comparison of the relative number of sources, sinks and isolated nodes in the standardized curricula

- US1-12.1: The students are able to identify jobs that use technology ([SCF⁺11]).
- US2-1.2: The students are able to select appropriate software to perform a variety of tasks ([CST17]).

• **sources:**

- AT-12.3: The students are aware that there are dangers such as malware, especially when using the Internet ([Dig13b]).
- AU-42.2: The students are able to use criteria for success, including sustainability considerations, to judge the suitability of their ideas, solutions and processes ([Aus13]).
- CH-40.1: The students are able to apply solution strategies to problems with devices ([Leh14]).
- DE-42.2: The students are able to explain that personal data can be processed with computer systems ([Ges19]).
- GB-12.4: The students are able to create a range of systems that accomplish given goals ([Ber15]).
- US1-36.2: The students are able to apply strategies for identifying simple software problems that may occur during use ([SCF⁺11]).

Tab. 5.6: Numbers of connected components

Number of	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Original curricula							
connected components	3	1	3	6	4	2	6
Standardized curricula							
connected components	3	1	3	6	5	2	7

- US2-25.4: The students are able to use data to communicate an idea ([CST17]).

- **isolated nodes:**

- CH-30: The students know the names of the document types they use ([Leh14]).
- GB-5: The students are able to discuss common uses of information technology beyond school ([Ber15]).
- US1-37: The students are able to identify that information is coming to the computer from many sources over a network ([SCF⁺11]).
- US2-22: The students are able to model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination ([CST17]).

A further metric to analyze the structure of a graph are the *numbers of connected components* (see definition 4.19). *Connected components* can show independent topics or competency paths. Tab. 5.6 shows the absolute numbers of connected components in the original and standardized curricula.

The only differences between the original and the standardized versions can be seen in the model for the English curriculum and the CSTA standards from 2017. It concerns the following original competencies with their subcompetencies from the EC/GB [Ber15]:

- GB-12.1: I can design a range of programs, systems and content that accomplish given goals.

- GB-12.1: I can design a range of programs that accomplish given goals.
- GB-12.2: I can design a range of systems that accomplish given goals.
- GB-12.3: I can design a range of content that accomplish given goals.
- GB-12.2: I can create a range of programs, systems and content that accomplish given goals.
 - GB-12.4: I can create a range of programs that accomplish given goals.
 - GB-12.5: I can create a range of systems that accomplish given goals.
 - GB-12.6: I can create a range of content that accomplish given goals.

It can be seen that the standardization process separated into topics of the design and creation of programs, systems and content. As programs and systems are connected to the development, the content is in most cases connected to the creation of digital documents. That means, the subcompetencies concerning programs and systems in most cases are independent from the creation of content. This leads to a separation and to two unconnected components. The following competencies and subcompetencies from the CSTA17/US are also concerned [CST17]:

- US1-3: The students are able to describe basic hardware and software problems using accurate terminology.
 - US1-3.1: The students are able to describe basic hardware problems using accurate terminology [3.1].
 - US1-3.2: The students are able to describe basic software problems using accurate terminology [3.2].

Here, a similar situation occurs like described before. The description of hardware and software problems are related, but non depends on the other one. This also results in a separation and two unconnected components.

Fig. 5.13 shows the normalized numbers of connected components to the number of overall competencies in the curricula. With the normalized values a comparison between the curricula is more meaningful.

It is obvious that these values are higher in the original curricula, as they contained less competencies before the standardization. A very low value and thus a high interconnection can be seen in the Australian curriculum which has one connected component, as it can be seen in Tab. 5.6. In the original versions the CSTA standards from 2017 and the GI standards have the highest normalized values and with six connected components also the highest number of connected components. Although the CSTA standards from 2017 have one more connected component in

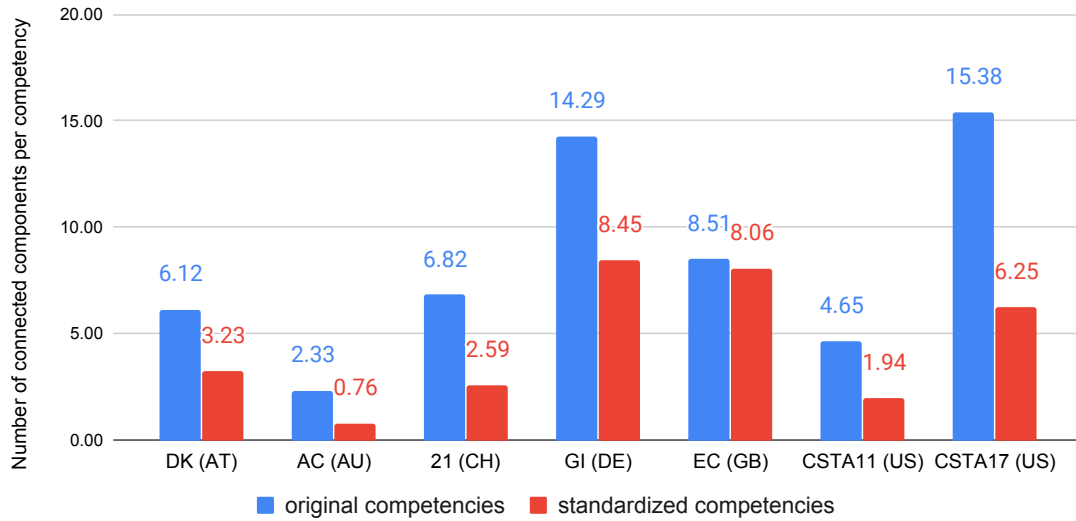


Fig. 5.13: Comparison of normalized numbers of connected components between original and standardized curricula

the standardized version, its normalized values are not the highest. The GI standards and the model for the English curriculum show higher normalized values, as their number of competencies is lower. That supports the assumption that the GI standards and the model for the English curriculum are slightly interconnected. In the context of curricula that means, the competencies belong to their topics but are independent from competencies in other topics.

5.5 Comparison of the Categories

5.5.1 Computer Science or Digital Literacy

As described in section 4.6.3, the first step of categorization is a survey with experts to categorize the original competencies of the curricula into *computer science* (CS) or *digital literacy* (DL). The overall results from that survey can be seen in Fig. 5.14. It represents a comparison of the numbers of competencies categorized into one of the three categories 'CS', 'DL', and 'Undecided' relative to the overall numbers of competencies in the particular curriculum. The results for the Austrian digital competency model (DK) show that the focus is clear on *digital literacy*. With only 4% in 'CS' and 4% in 'Undecided', 'DL' has 92% of the competencies. Also the curriculum from Switzerland (21) and the CSTA standards from 2011 (CSTA11) lay their focus on *digital literacy*, though the numbers of 'CS' rated competencies are with 23% in both curricula a bit higher. With 74% in the 21 and 74% in the

CSTA11 also the values for 'DL' are very similar. Only the number of 'Undecided' ratings is in the *21* with 5% higher, then the 2% in the *CSTA11*. The Australian curriculum (AC) is with a distribution of 45% in 'CS', 50% in 'DL', and 5% in 'Undecided' very balanced. And the only model with more competencies in 'CS' are the CSTA standards from 2017 (CSTA17) with 51% in 'CS', 46% in 'DL', and 3% in 'Undecided' [PKB19]. The full number can also be found in Tab. 5.7.

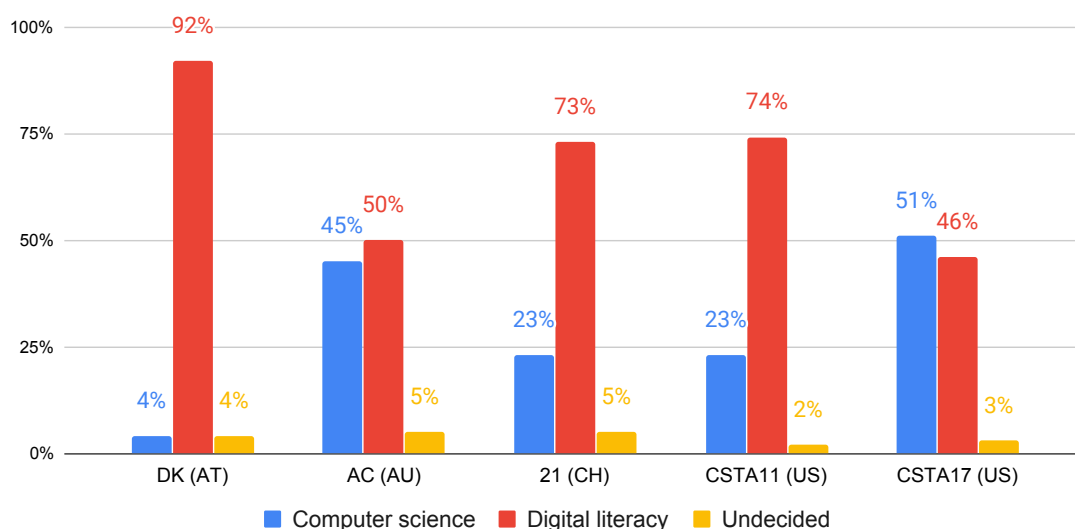


Fig. 5.14: Comparison of the focus from the five analyzed curricula

Additionally, *strong agreements* are analyzed. A '*strong agreement*' is given, if at most one single expert disagrees with the other categorizations. The percentage of strong agreements for each category can be found in Fig. 5.15. It shows that the decisions in the *DK* were very clear because the experts agreed in 88% of the ratings for 'DL'. Also, within 43% in the *21* and 37% in the *CSTA11* the categorization into 'DL' was clearer. For competencies in 'CS' the experts had different opinions. Only in 7% of the cases in the *21* strong agreements were reached. No strong agreements in this category were found in the *CSTA11*. In the *AC* and the *CSTA17* the experts agreed more often in the category 'CS' with 32% (*AC*) and 23% (*CSTA17*). Here, the number of strong agreements in 'DL' are with 9% (*AC*) and 13% (*CSTA17*) lower [PKB19]. Again, the number is shown in Tab. 5.7.

Following the guidelines of Fleiss [Fle81] a value of 0.43 for the *inter-rater agreement* was determined. This value reflects the reliability of the agreement when categorized by a given number of experts. A value of 0.43 shows a '*fair to good*' agreement in this survey [PKB19].

A more detailed analysis of three selected curricula is already described by Pasterk and Bollin in the following words [PB17a, p. 4]: To accomplish our goal

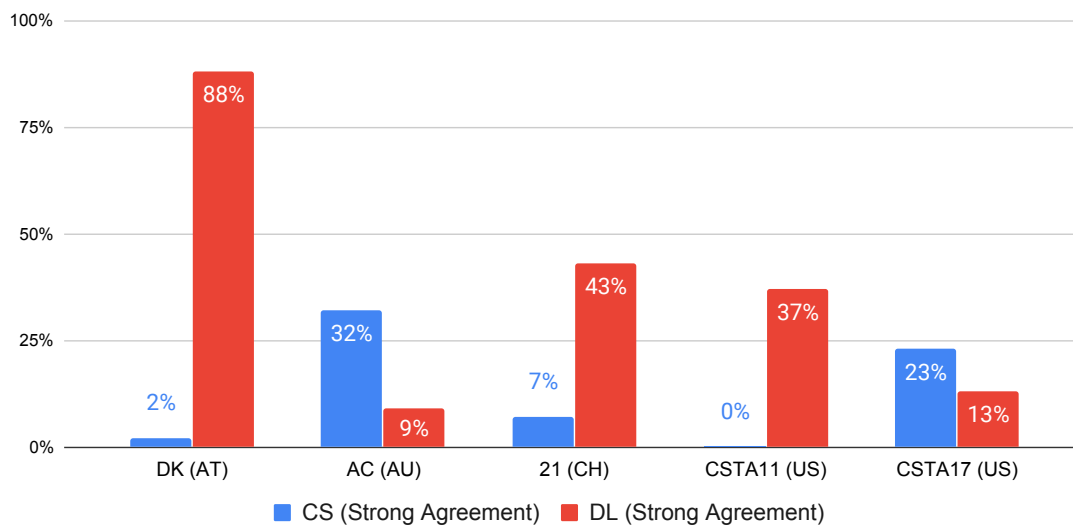


Fig. 5.15: Comparison of the strong agreements in the experts' decisions

Tab. 5.7: Percentage of the categories *computer science* and *digital literacy* in curricula

Category	DK (AT)	AC (AU)	21 (CH)	CSTA (US)	
				2011	2017
Computer Science (CS)	4%	45%	23%	23%	51%
Computer Science (CS) strong agreement	2%	32%	7%	0%	23%
Digital Literacy (DL)	92%	50%	73%	74%	46%
Digital Literacy (DL) strong agreement	88%	9%	43%	37%	13%
Undecided	4%	5%	5%	2%	3%

of giving an overview of the focus of the selected educational models, we represent the results from the expert survey as graphs with different colored nodes. For each curriculum or competence model three graphs with different coloring are presented. The first and the second graphs show the separate numbers of votes for *CS* and *DL*. So the colors of the nodes in the first graph represent how many experts chose *CS* or *Both* for each of the corresponding competence or learning objective, and the

second graph does the same for *DL*. This is of interest because the two categories are often related together, so it can occur that even if more experts chose one category for one competence or learning objective some would classify it to the other category. Similar cases can be identified by a comparison of the first and the second graph. The third graph divides the nodes into '*CS*', '*Rather CS*', '*Draw*', '*Rather DL*' or '*DL*'. A node is classified as '*CS*' or '*DL*', if more than 75 percent of the experts chose *CS* or *DL* for the corresponding competence or learning objective. If more than 50 but less than 75 percent of the experts chose *CS* or *DL*, the node is classified as '*Rather CS*' or '*Rather DL*'. If exactly the half of the experts voted for *CS* and the other half for *DL*, then the node is displayed to be '*Draw*'. With nine experts this can only happen, if at least one expert voted with *Both* at the corresponding competence or learning objective because it is counted for *CS* and *DL*. In this section the results for each analyzed educational model are presented and discussed.

Pasterk and Bollin continue as follows [PB17a, p. 4]: The separate results for each of the two categories *CS* and *DL* of the Australian curriculum are presented in Fig. 5.16 and Fig. 5.17. Fig. 5.16 shows that at least eight experts voted for ten learning objectives to belong to *CS*. Further, one learning objective was classified by at least six experts into *CS*, four learning objectives by at least four experts, three learning objectives by at least two experts, and two learning objectives by one expert. Only two learning objectives have not been classified into *CS* by any expert. In comparison to Fig. 5.17 six learning objectives were never classified to belong to *DL*. Four learning objectives were classified by at least eight experts into *DL*, six learning objectives by at least six experts, three learning objectives by at least four experts, one learning objective by at least two experts, and two learning objectives by one expert. These results don't indicate a focus on one of the two categories, but it seems that the experts agreed on the classification of learning objectives into *CS*, whereas their opinions concerning *DL* were a bit more divided. For this comparison it has to be considered that some nodes are classified by at least eight experts into one category, but have also e.g. four votes for the other category. An example for this case is indicated in Fig. 5.16 and Fig. 5.17 by a green cycle. That can occur, if several experts chose *Both* for the corresponding learning objectives and several others chose one of the categories '*CS*' or '*DL*'.

Pasterk and Bollin go on with their analysis of the Australian curriculum [PB17a, p. 4]: Fig. 5.18 shows an overall comparison of the Australian curriculum. Exactly the same amount of the learning objectives (ten) were classified into *CS* and into *DL* by the majority of the experts. For *CS* nine learning objectives were classified into '*CS*' because over 75 percent of the experts chose *CS* for the corresponding learning objectives, and only one into '*Rather CS*' which means that between 50 and 75 percent of the experts chose *CS*. On the other hand only three learning objectives are classified into '*DL*' because over 75 percent of the experts voted for *DL*, and

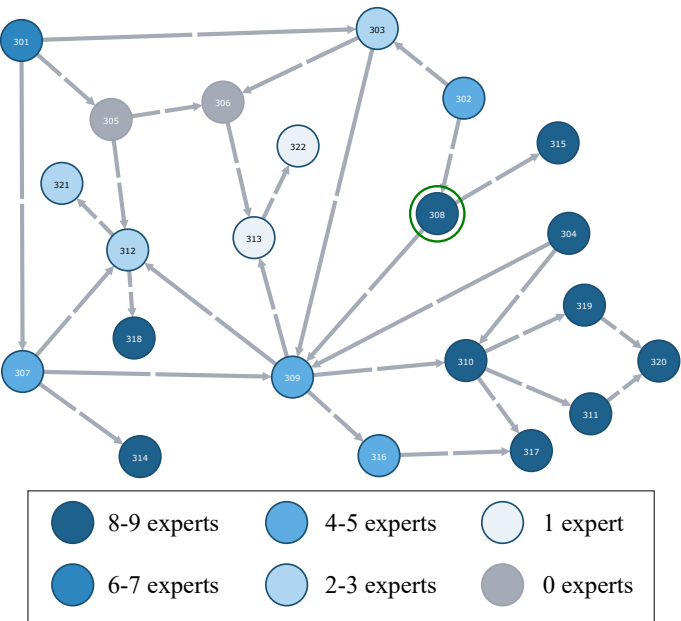


Fig. 5.16: The distribution of experts' choices for CS for the learning objectives from the Australian curriculum [PB17a]

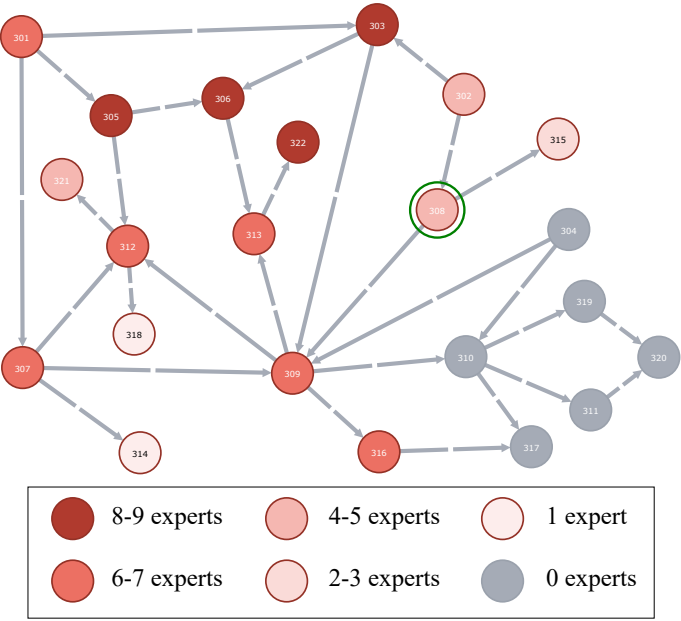


Fig. 5.17: The distribution of experts' choices for DL for the learning objectives from the Australian curriculum [PB17a]

seven into 'Rather DL' because between 50 and 75 percent of the experts voted for *DL*. For two nodes both categories got the same amount of votes that is why they are classified as 'Draw'. These results show that the Australian curriculum as a balanced number of learning objectives of each category and has no clear focus.

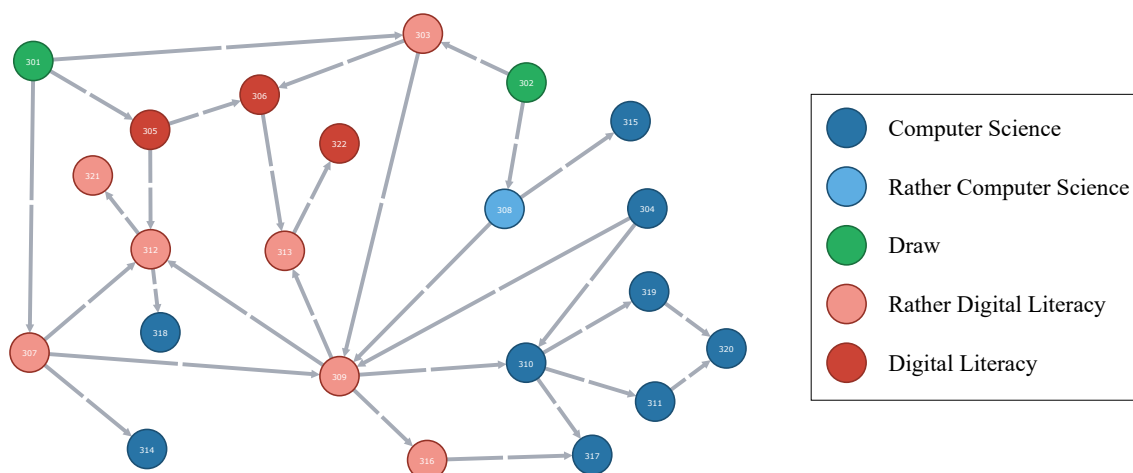


Fig. 5.18: A comparison of learning objectives related to *CS* or *DL* from the Australian curriculum [PB17a]

Pasterk and Bollin analyze the curriculum from Switzerland in the following words [PB17a, p. 5]: For the curriculum from Switzerland the separate results are presented in Fig. 5.19 and Fig. 5.20. Fig. 5.19 shows that the distribution of choices for *CS* is limited to a few nodes. Nine nodes were classified by at least eight experts into *CS*. On the other hand there are 17 nodes that don't have any vote for *CS*. Fig. 5.20 shows also a clear focus on *DL* because 25 nodes are classified by at least eight experts into *DL*, whereas only three nodes include no vote for *DL*. The overall comparison in Fig. 5.21 shows only five nodes are classified to 'CS' and six to 'Rather CS', whereas 22 nodes belong to 'DL' and ten to 'Rather DL'. And one node is classified into 'Draw'. Following the decisions of the experts the focus of the curriculum from Switzerland is on *DL*.

Pasterk and Bollin end with their analysis of the Austrian competency model [PB17a, p. 5]: The separate results for the Austrian Digital Competence model are presented in Fig. 5.22 and Fig. 5.23. Fig. 5.22 shows that only one competence was classified into *CS* by at least eight experts. On the other hand 25 competences had no single vote for *CS* by any expert. Fig. 5.23 shows the dominance of *DL* in this model. From overall 49 competences 43 were classified by at least eight experts into *DL*. The overall comparison of the Austrian model presented in Fig. 5.24 show that there are only two nodes classified into 'Rather CS' and two into 'Draw'. The rest belongs with 38 nodes to 'DL' and with seven nodes to 'Rather DL'. As the Austrian

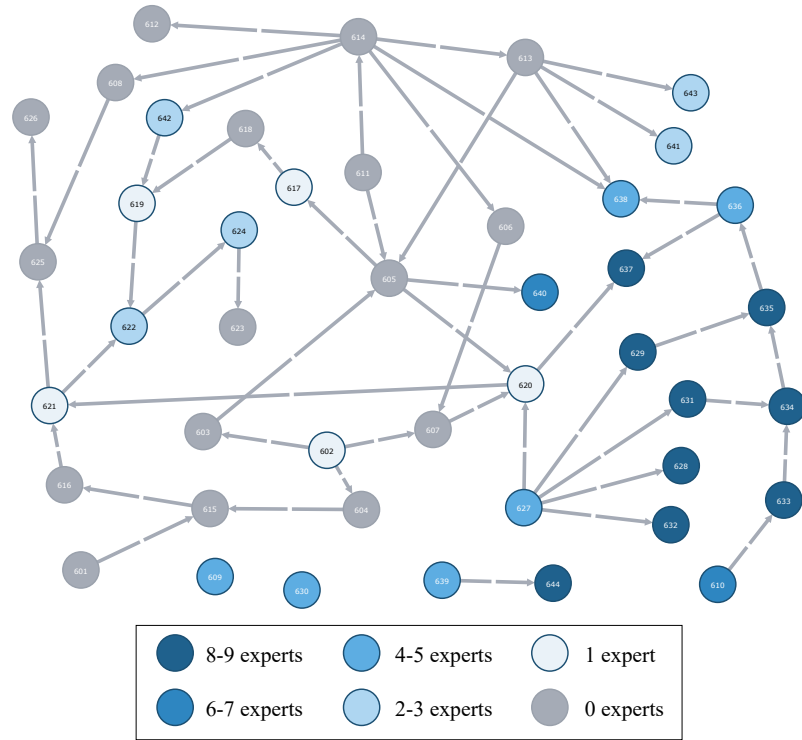


Fig. 5.19: The distribution of experts' choices for CS for the competence levels from the curriculum in Switzerland [PB17a]

model is a model for Digital Competence it has a strong focus on *DL*.

It has to be mentioned that during the expert rating only the original competency formulations from the curricula were categorized and the standardized formulations were not considered.

5.5.2 Distribution of Categories

During the coding process, also the original versions were coded and categorized. As the coding was done per phrases and not per competency, it happened that different codes were present in one competency. In some cases this led to questions in categorization, as the codes of one competency belonged to more than one category. This was resolved by the numbers of occurring codes. If the number of codes from one category was higher than the others, the competency was categorized into this category. If this was not the case, it was decided by the researchers which code had a higher weight. For the original competencies, Fig. 5.25 shows the relative distribution of competencies over the five categories *programming and algorithms*, *data representation*, *digital infrastructure*, *digital applications*, and *human factors and ethics* (see section 3.2). The Austrian competency model (DK) contains over

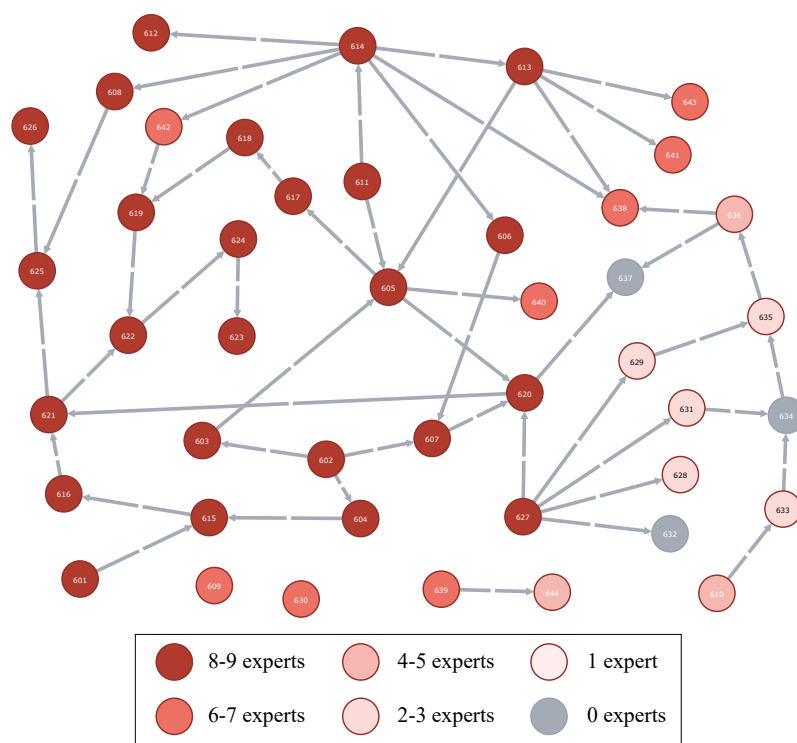


Fig. 5.20: The distribution of experts' choices for DL for the competence levels from the curriculum in Switzerland [PB17a]

40% of the competencies from the category *human factors and ethics* followed by *digital applications* with over 28%. In the Australian curriculum the most frequently occurring category is *programming and algorithms* with nearly 35% followed by *human factors and ethics* with 30%. A very close result for the most frequent category can be seen in the curriculum from Switzerland with 30% for *digital applications* and 27% for *digital infrastructure*. *Digital infrastructure* is by far the most frequent category in the GI standards from Germany with 43%, followed by *human factors and ethics* with 21%. In the model from Berry for the English curriculum the category *programming and algorithms* shows with 53% the highest value not only within the curriculum but of all compared curricula. It is followed by *digital applications* with 26%. The most frequent category in the CSTA standards from 2011 is with 33% *human factors and ethics*, followed by *digital applications* with 26%. Also in the CSTA standards from 2017 the category *programming and algorithms* is very popular with 46%. The second most frequent category is *human factors and ethics* with 21%.

Taking a more detailed look at the categories and their occurrence shown in Fig. 5.26, *programming and algorithms* presents more than 20% in four curricula.

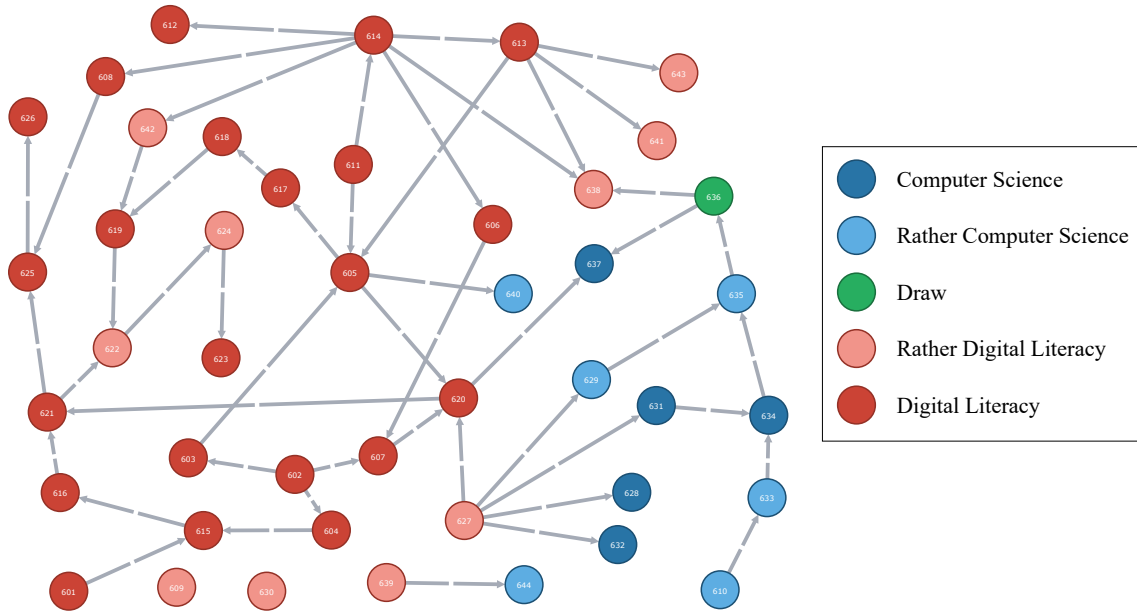


Fig. 5.21: A comparison of competence levels related to *CS* or *DL* from the curriculum from Switzerland [PB17a]

The category *data representation* shows in no curriculum a higher value than 12% and in the model for the English curriculum no competency of this category occurs. Only in two curricula the category *digital infrastructure* presents more than 20% and in two curriculum it shows under 10%. *Digital applications* shows in four curricula more than 25% and only in one under 10%. The most frequent category over all curricula is *human factors and ethics* which shows in all curricula over 17% and in three curricula over 30%.

It can be assumed that with the standardized competency formulations explained in section 4.6.2, the above described results do not change a lot. Some points of interest are visible though as Fig. 5.27 shows. For instance, the category *programming and algorithms* increased the frequency in four curricula for at least 5%. This leads to two possible interpretations: first, the competencies from this category contained several subcompetencies and were separated more often than others. Or second, some subcompetencies from other competencies moved to this category. The model from Berry for the English curriculum contains after standardization 3% competencies of the category *data representation*. Before standardization it had 0%. This are in absolute numbers two competencies in this category. In this case it is obvious that subcompetencies have been moved from other categories into this category. The standardization of the formulations additionally solved the problem of more codes within one competency. After this process it happened very rarely that one competency included more codes. And in these very rare cases the codes belonged

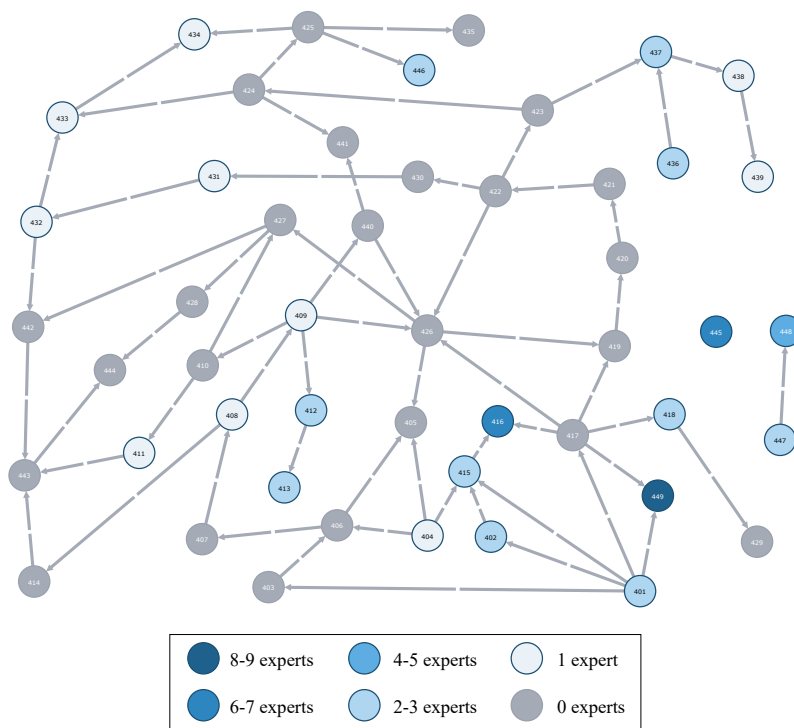


Fig. 5.22: The distribution of experts' choices for CS for the competences from the Austrian competence model [PB17a]

to the same category. So, the process was simplified by the prior standardization.

There are no notable changes visible in the occurrence of the categories after standardization and therefore they are not discussed.

5.6 Conclusion

This chapter discusses several aspects of comparing the computer science related curricula for primary education. It starts with the related work in analyzing curricula representing graphs and defines basic metrics and centrality measures to compare the curricula. Further, the results from the comparison of the basic metrics, the identification of central competencies, the analysis of the graph structure, and the categorization are discussed. These show that the original curricula are very similar in their numbers of competency, but after the standardization process the highest and the lowest numbers differ by 69 competencies (Australian curriculum with 131, and the model for the English curriculum with 62 competencies). This reflects the fact that the formulations in the curricula are different in their use of verbs and objects. Some include more than one verb to describe the actions the students should

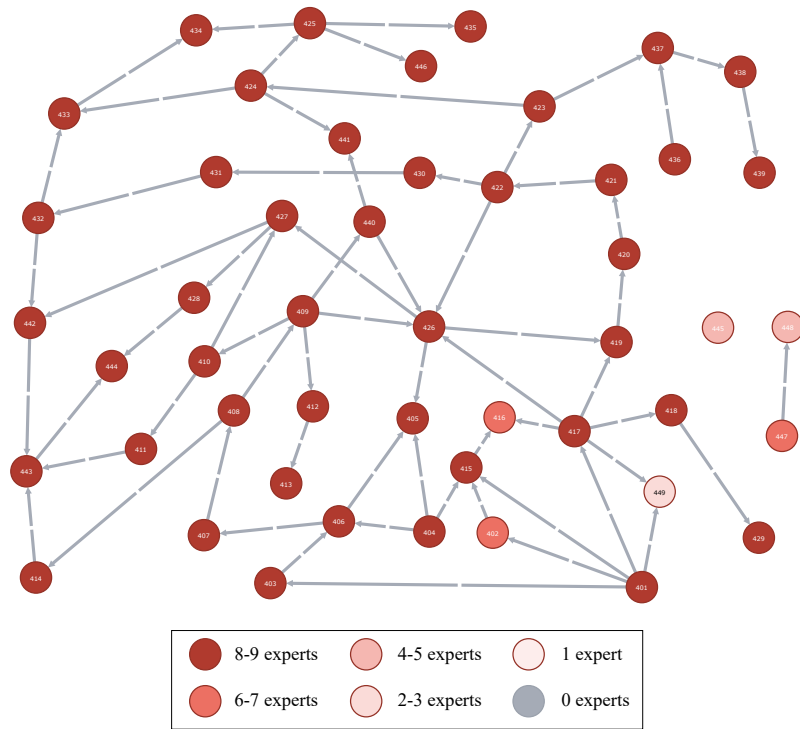


Fig. 5.23: The distribution of experts' choices for DL for the competences from the Austrian competence model [PB17a]

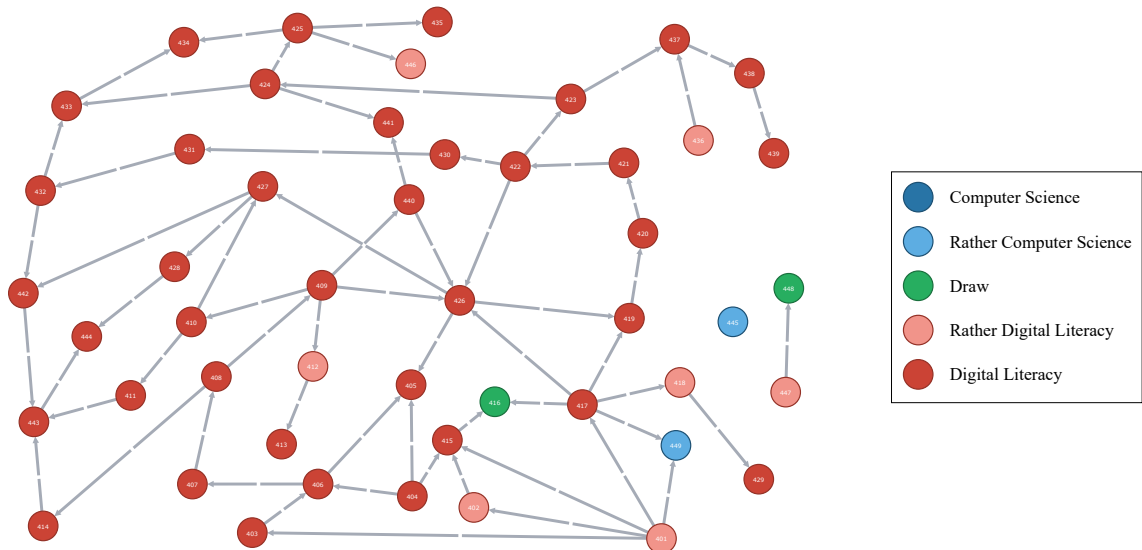


Fig. 5.24: A comparison of competences related to *CS* or *DL* from the Austrian competence model [PB17a]

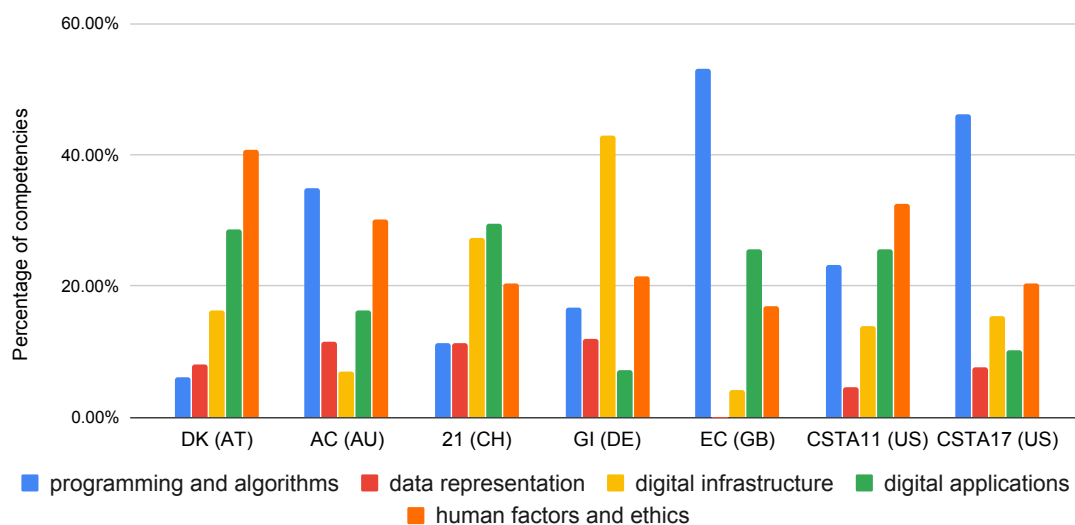


Fig. 5.25: Relative distribution of original competencies over the five categories

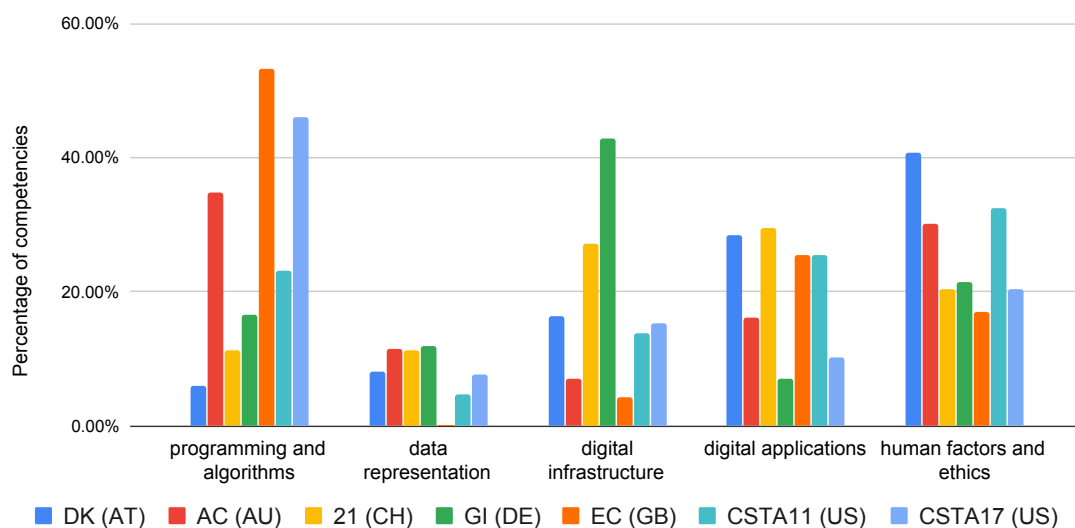


Fig. 5.26: Comparison of the five categories based on original competencies

be able to do with sometimes more than one object to reach the level. Also, the numbers of relations per competency show differences between the curricula. The competencies from the original CSTA standards from 2011 seem to be very inter-related, as this curriculum shows the most relations per competency. The lowest number of relations per competencies in the original curricula can be found in the GI

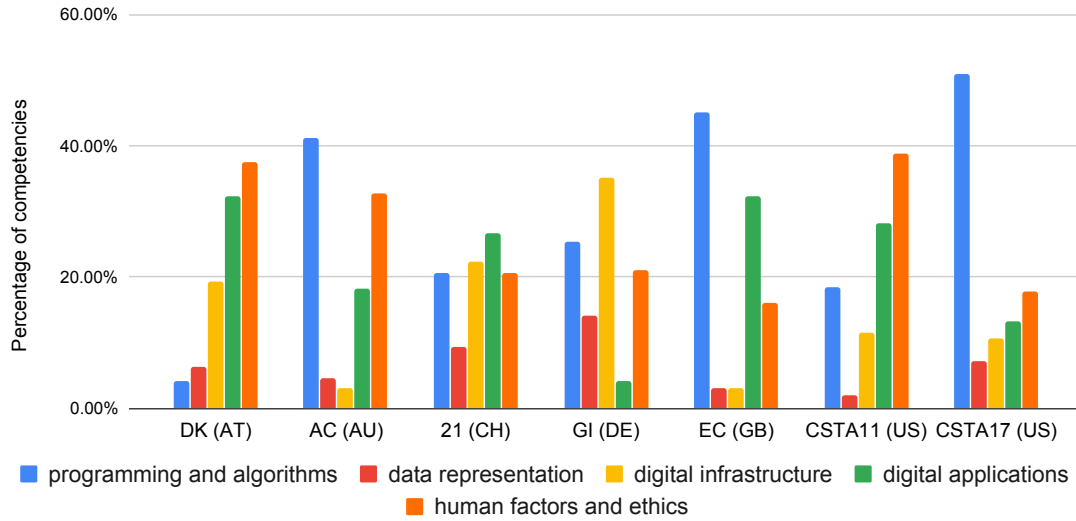


Fig. 5.27: Relative distribution of standardized competencies over the five categories

standards and the model for the English curriculum. This conclusion is supported by the density values. These basic values open the opportunity of comparing curricula. Additionally, centrality measures like *degree centrality*, *betweenness centrality*, and *PageRank* provide further possibilities. They are used to identify central nodes in the graph representations and thus central competencies in the curricula. The results show that competencies with a high *in-degree* are basic competencies which are necessary for other competencies, and that competencies with a high *out-degree* have a connective character as they bring competency paths together. High values in the *betweenness centrality* are a bit harder to interpret, as the structure of the graphs has to be considered. In some cases those competencies also show a connective character but often only for one larger topic. *PageRank* adds information about basic competencies considering transitivity. The discussed competencies with the highest *PageRank* values in the curricula are all on a very basic level and often reflect the focus of the corresponding curriculum. The structure of the curricula is analyzed by their number of special nodes like *sinks*, *sources*, and *isolated nodes* as well as the number of *connected components*. *Sinks* and *sources* represent on the one hand competencies which can be starting, respectively ending points, and on the other hand reflect the number of started, respectively ended topics or competency paths. *Isolated nodes* are competencies which are not connected to others and often address topics which are important in computer science but do not have direct relations to the other topics. The number of *connected components* also shows how interconnected the topics or competency paths in a curriculum are. Here, the GI standards and the CSTA standards from 2017 show the highest values. That means,

their topics are more independent from the other topics. The graph representing the Australian curriculum consists of one *connected component* which means that all of its topics are interconnected. The categorization of the competencies into computer science and digital literacy is used to determine the general foci of the curricula. The results from this show that the Austrian competency model, the curriculum from Switzerland and the CSTA standards from 2011 have a trend towards digital literacy. The competencies from the curriculum from Australia and the CSTA standards from 2017 are more evenly distributed between the two categories. Also the more detailed categorization of the competencies into the five computer science education categories shows that the Austrian competency model and the CSTA standards from 2011 show most competencies in the category *human factors and ethics*, whereas the Australian curriculum, the model for the English curriculum, and the CSTA standards from 2017 show most competencies in *programming and algorithms*. The focus of the curriculum from Switzerland can be found in the category *digital applications* but very closely followed by *digital infrastructure*, and the GI standards have a trend towards *digital infrastructure*. The following chapter presents the results from the analysis of the combined *GGBMC* and describes the calculation and optimization of *learning paths* in it.

6. IDENTIFICATION OF CENTRAL COMPETENCIES AND LEARNING PATHS IN THE GGBMC

6.1 Introduction

In the previous chapter the results from the analysis and comparison of the curricula and their graph representations are discussed. For the *Generic, Graph-Based Model for Competencies (GGBMC)* which combines the single curricula over *intersector nodes* to one generic model, similar analysis methods are applied. The values for the basic metrics and results concerning the *intersector nodes* are presented in section 6.2. In section 6.3 the results for the centrality measures are discussed to identify central nodes in the *GGBMC*. Section 6.4 takes a look at so called *learning paths* and methods to determine and optimize them in the *GGBMC*.

6.2 Analysis of GGBMC

6.2.1 Basic Measures

After the *standardization* and the *categorization*, the curricula are combined to the *Generic, Graph-Based Model for Competencies (GGBMC)*. Here, the *intersector nodes* play an important role, as they collect similar competencies of different curricula. The result can be seen in Fig. 6.1 showing the *GGBMC* in a multicircle layout. The red nodes forming the circle in the middle represent the *intersector nodes*. Each other larger circle is formed by the competencies from one curriculum. The smaller circles within the curricula bundle the competencies by sub-categories.

In this and the following sections the *GGBMC* is analyzed and the *basic measures*, including information about the *intersector nodes*, as well as *centrality measures* are presented.

Tab. 6.1 shows the results for basic measures in the *GGBMC*. Overall it contains 538 nodes, 691 relations, and has thus a density of 0.238. In the comparison to simple sum or average values from the single curricula, differences can be recognized. Because of the combination of competencies to *intersector nodes*, the number of overall competencies is smaller than the sum of all competencies from the single curricula. This also means that the number of relations is lower in the *GGBMC*. The *GGBMC* contains 20 *sinks* and 45 *sources* less than the sum of the single curricula.

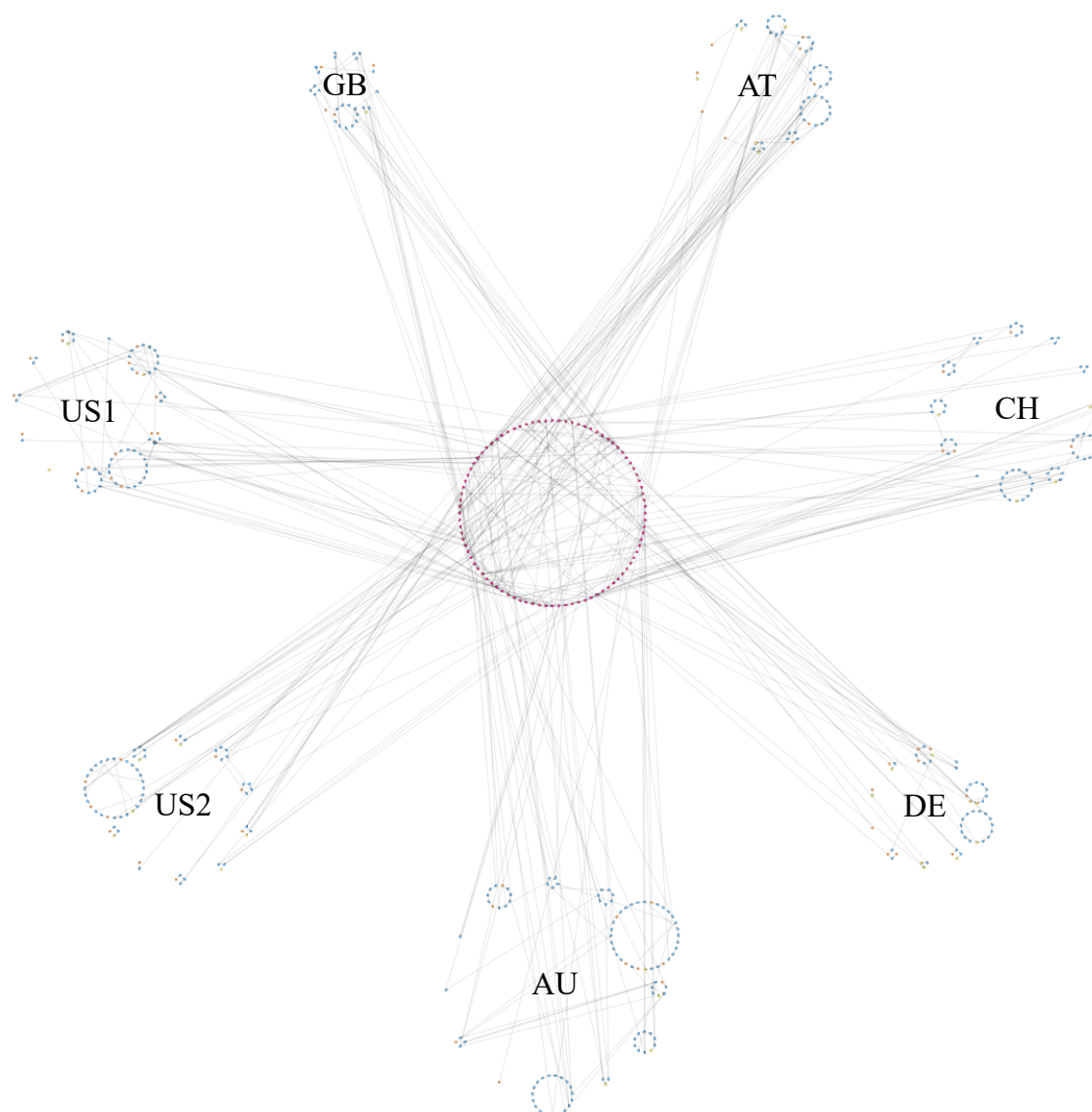


Fig. 6.1: The *GGBMC* in a multicircle layout

That means that 20 *sinks* and 45 *sources* are part of *intersector nodes*. The number of *isolated nodes* is the same, so none of them is combined to an *intersector node*. Also there are, with a number of 14, 13 *connected components* less in the *GGBMC* than in the sum of all single curricula. This shows that the combination with *intersector nodes* also connects components which are independent in the single curricula. So, *intersector nodes* have a meaningful impact on the structure and the interconnection of the *GGBMC*. The *density* of the *GGBMC* is (with 0.238) by far lower than the average value of all single curricula. This large difference can be

Tab. 6.1: Results for basic measures of the GGBMC

	GGBMC	Sums of single curricula
number of nodes	538	688
number of relations	691	722
number of sinks	31	51
number of sources	71	116
number of isolated nodes	4	4
number of connected components	14	27
	GGBMC	Average of single curricula
density	0.238	1.131

explained by the fact that larger networks are less densely connected [Pre12].

For a comparison, the average, the highest, and the lowest numbers of the *sinks*, *sources*, *isolated nodes*, and *connected components* relative to the numbers of competencies from the single curricula are taken and presented in Fig. 6.2. Comparing them to the numbers of the *GGBMC* a trend is visible. It can be seen that the relative numbers of *sinks*, *sources*, and *connected components* of the *GGBMC* are located between the average and the lowest values of the single curricula. As the existing *sinks* and *sources* cannot disappear, there is only one interesting explanation for that. Some *sinks* and *sources* are combined to *intersector nodes* and, as there are only four *intersector nodes* which are also *sinks* and three which are also *sources*, some of them have outgoing respectively incoming relations in the *GGBMC*. That means that in some cases a competency which is a *sink* in one curriculum, has in form of its *intersector node* dependencies to other competencies from other curricula. Or in case of a *source* in one curriculum other competencies or *intersector nodes* are dependent on it in the *GGBMC*. As the *sinks* mark starting points in the curricula this would mean that some of the found starting points in curriculum *A* can have prerequisites in a curriculum *B*. But, these prerequisites are missing in curriculum *A*.

Comparable cases can occur for the *sources* representing competencies which no other competencies depend on. It can be assumed that a curriculum *A* with a node *s*, which is a *source* in *A*, does not go that deep into a topic as curriculum *B* does which also includes *s* but with incoming dependencies.

Considering the *isolated nodes* there is no big difference between the relative

number of the *GGBMC* and the average numbers from the single curricula. The relative number of *connected components* is lower than the average but higher than the lowest number in the single curricula.

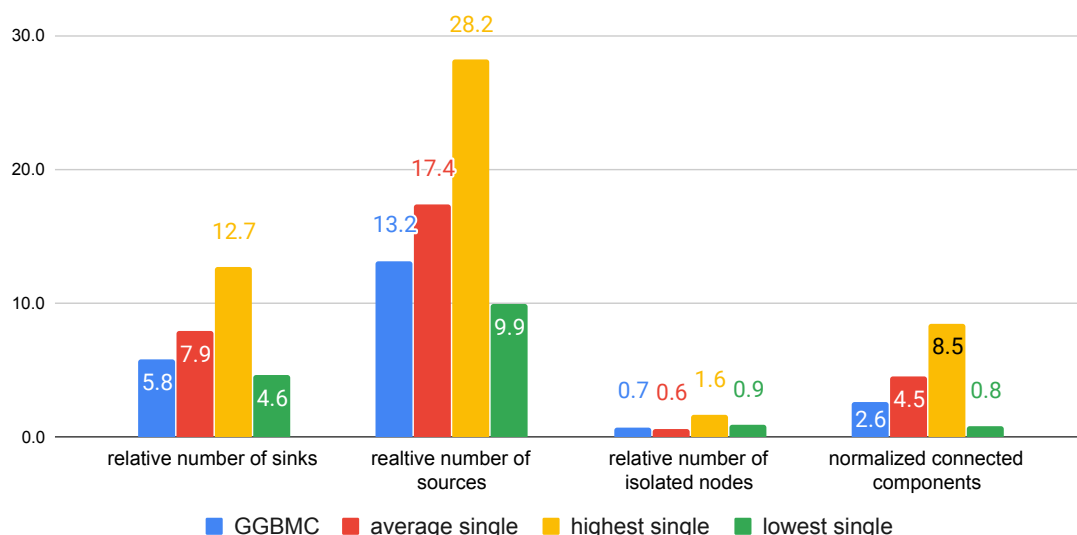


Fig. 6.2: Comparison of relative numbers of sinks, sources and isolated nodes

It is visible that the *intersector nodes* play an important role in the *GGBMC*. Therefore, they are analyzed in more detail in the following section.

6.2.2 Analysis of Intersector Nodes

Overall 238 competencies are combined to 88 *intersector nodes*. That means, 16% of all nodes in the *GGBMC* are *intersector nodes* and 34.6% of all competencies from the single curricula are combined to *intersector nodes*. Each curriculum shows a different amount of competencies included in *intersector nodes*. This can be seen in Fig. 6.3. From the CSTA standards from 2017 and the curriculum from Switzerland the highest number of competencies have been combined with others. This means, most similarities between their competencies and others are to be found. The by far lowest number of competencies, and with that similarities to other competencies, are to be found in the GI standards followed by the Australian curriculum.

The distribution of *intersector nodes* as well as competencies over the five categories for computer science education can be found in Fig. 6.4. It shows that overall most competencies belong to the category *programming and algorithms* followed by *human factors and ethics*. The lowest number of competencies still is present in the category *data representation*. The distribution of the *intersector nodes* shows very close numbers between 19 and 22 in four of the five categories, with the highest value

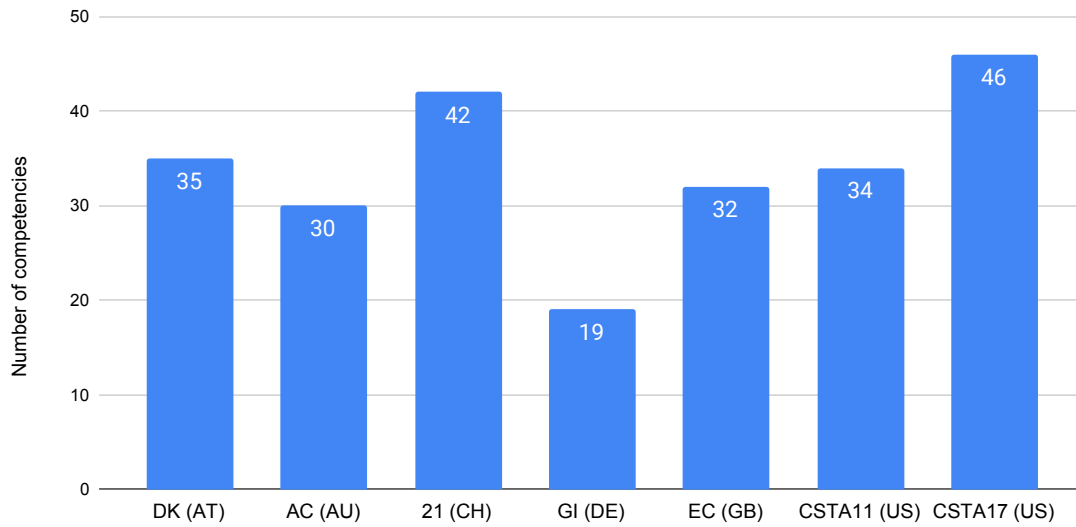


Fig. 6.3: Numbers of competencies in each curriculum combined to intersector nodes

in *digital applications*. The exception is the category *data representation*, where only 6 *intersector nodes* belong to.

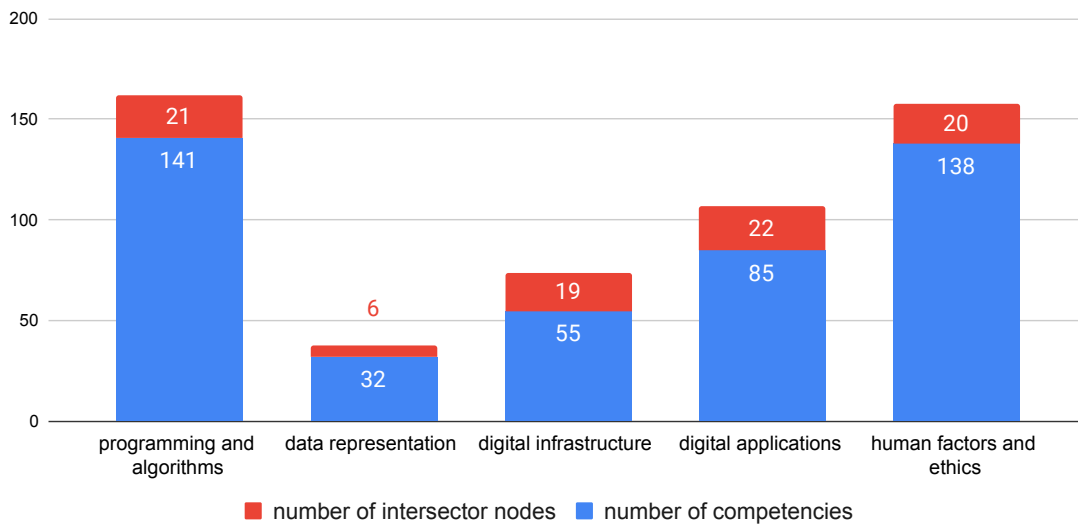


Fig. 6.4: Comparison of numbers of competencies and intersector nodes in the categories

Some of the *intersector nodes* also represent special nodes, as they replace common competencies. None of them is an *isolated node*, but the following four *inter-*

sector nodes are identified to be *sinks*. *Intersector nodes* are presented here with their ID, including the prefix 'INT', their number in the category they appear in and the abbreviation of the category, and by their bundled competencies from the single curricula:

INT-001-DI:

- AT-20.1: The students are able to start a computer [Dig13b].
- CH-11.1: The students are able to switch devices on [Leh14].

INT-001-PA:

- AT-47.1: The students are able to understand simple instructions [Dig13b].
- CH-10.1: The students are able to recognise formal instructions [Leh14].

INT-009-DI:

- DE-13.0: The students are able to name the components of computer systems using the technical language of computer science [Ges19].
- US2-2.2: The students are able to use appropriate terminology in describing the function of common physical components of computing systems (hardware) [CST17].

INT-018-DA:

- CH-9.0: The students are able to sort things according to their own chosen properties so that they can find an object with a certain property more quickly [Leh14].
- US1-5.1: The students are able to understand how to arrange (sort) information [SCF⁺11].
- US1-5.2: The students are able to understand useful order, such as sorting students by birth date [SCF⁺11].

In the case of dependency relations, *sinks* are competencies to start with as they do not require previous knowledge or skill in the topic. This is also reflected by the four *intersector nodes* without outgoing relations. Also, four *intersector nodes* show no incoming dependencies and are therefore *sources*:

INT-005-DR:

- AU-30.0: The students are able to explain how digital systems use whole numbers as a basis for representing a variety of data types [Aus13].

- US1-22.0: The students are able to demonstrate how a string of bits can be used to represent alphanumeric information [SCF⁺11].

INT-013-DI:

- CH-40.1: The students are able to apply solution strategies to problems with devices [Leh14].
- US2-21.1: The students are able to determine potential solutions to solve simple hardware problems using common troubleshooting strategies [CST17].

INT-014-DI:

- CH-40.2: The students are able to apply solution strategies to problems with programs [Leh14].
- US2-21.2: The students are able to determine potential solutions to solve simple software problems using common troubleshooting strategies [CST17].

INT-018-DI:

- DE-37.2: The students are able to describe how data is transferred on the Internet using fixed arrangements (protocols) [Ges19].
- GB-10.2: The students are able to explain how computer networks can provide multiple services such as the World Wide Web [Ber15].

These *intersector nodes* represent topics which are important to know, but are not directly necessary to reach other competencies. For example, problem solving for hardware or software problems (see INT-013-DI and INT-014-DI) are important competencies, but they are not essential on the way to learn programming or data organization.

6.3 Identification of Central Learning Outcomes in the GGBMC

6.3.1 Centrality Measures

Similar to the process in the single curricula, also during the analysis of the *GGBMC*, different centrality measures are used to identify central competencies. At first the *degree* values are discussed as they are the basis for the *degree centrality*. In Fig. 6.5, the highest normalized values of *degree*, *out-degree*, and *in-degree* from the *GGBMC* and the average, the highest, and the lowest values of the single curricula are compared. It can be seen that all values of the *GGBMC* are close to the lowest

values of the single curricula. In case of the maximum values of the *out-degree* the value of the *GGBMC* is even much lower than the lowest value from the single curricula. The highest *degree* values in the *GGBMC* are 16 for the overall *degree*, 9 for *out-degree*, and 10 for *in-degree*. These values are a bit higher than the highest *degree* values in the single curricula, but normalized to the number of competencies, which are partially ten times as high as the numbers of competencies in single curricula, they get very low.

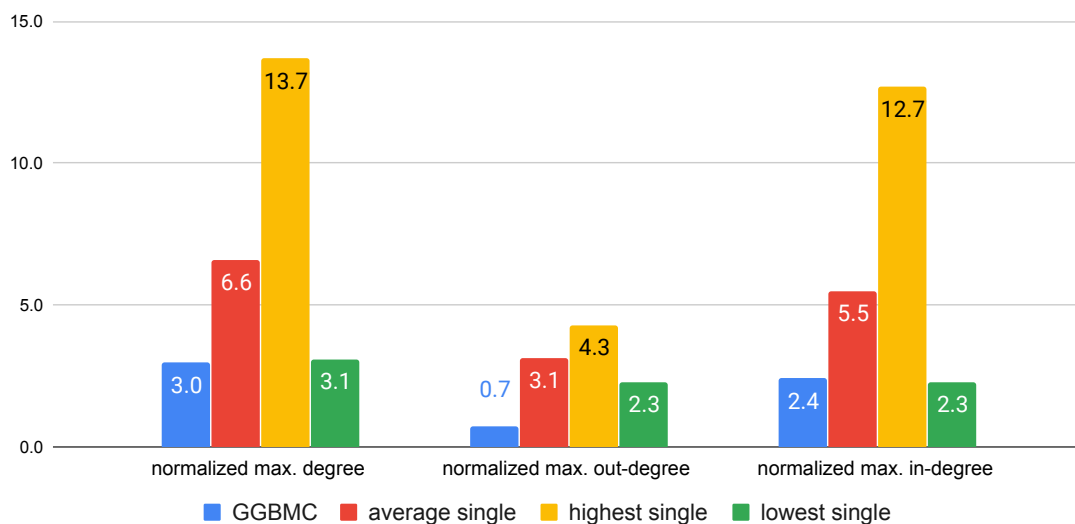


Fig. 6.5: Comparison of normalized degree, in-degree, and out-degree values

The node with the highest *degree* value of 16 is the following *intersector node* with its bundled competencies:

INT-009-HF:

- AT-27.0: The students are able to use networks to communicate [Dig13b].
- AU-13.3: The students are able to safely communicate ideas and information online [Aus13].
- CH-8.0: The students are able to use the media to maintain existing contacts and exchange information [Leh14].
- US1-1.6: The students are able to communicate electronically with others with support student partners [SCF⁺11].

Here, competencies from four different curricula are combined to an *intersector node*. They represent the ability to communicate with the help of technologies and

are therefore a very central competency in the *GGBMC*. The value of 10 for the *in-degree* shows that other competencies more often depend on it than it depends on others. With a value of 9 the following *intersector node* shows the highest *out-degree* and additionally the second highest overall *degree* of 12:

INT-014-DA:

- AT-46.1: The students are able to collect data [Dig13b].
- AU-18.1: The students are able to collect different data when creating information and digital solutions [Aus13].
- GB-12.7: The students are able to collect data [Ber15].
- US1-28.1: The students are able to gather data using a variety of digital tools [SCF⁺11].

The combination of these four competencies from four different curricula is obvious as they cover the same topic and use similar verbs. Following the nodes *out-degree* it depends on nine other competencies and three competencies depend on it. That means it connects nine independent paths which confirms it a central meaning for the *GGBMC*.

The highest *in-degree* value of the *GGBMC* still occurs in the CSTA standards from 2011 with the already mentioned competency [3] (*The students are able to use technology resources (e.g. puzzles, logical thinking programs) to solve age-appropriate problems.* [SCF⁺11]) and an *in-degree* value of 11. Additionally, this is the single competency with the highest overall *degree* of 12.

With an *out-degree* value of 3 the first competencies, so non *intersector nodes*, show up. These are e.g. from the Australian curriculum AU-31.1, and AU-32.1, from the CSTA standards from 2011 US1-38.0, US1-2.4, and US1-15.1, or from the curriculum from Switzerland CH-5.1, CH-20.1, and CH-38.0.

The results for the *betweenness centrality* in the *GGBMCs* show that the *intersector nodes* have a very connective character. This is indeed the idea behind these nodes. So, the 15 highest values of *betweenness centrality* are found in *intersector nodes*. The following *intersector node* with its combined competencies has the highest values:

INT-017-HF:

- AT-9.2: The students know how they should behave in a given case concerning the risks associated with the use of information technologies [Dig13b].
- GB-6.4: The students are able to explain where to go for help and support when pupils have concerns about content [Ber15].

This *intersector node* contains important competencies which are required in different situations and scenarios. It also reflects very basic abilities in the use of information technology. Nonetheless, previous knowledge is necessary to reach this level as students need to interpret content and evaluate the risks and concerns. So, the connective character is definitely given. The competency with the highest *betweenness centrality* appears in the curriculum from Switzerland and connects two *intersector nodes* which can be an explanation for the high value.

CH-43.2: The students are able to distinguish between local devices, local networks and the Internet as storage locations for public data [Leh14].

Nonetheless, it is an important topic and necessary for several other competencies. In addition, it depends on previous knowledge of devices, networks and the Internet, which provides it a connecting position.

Results from *PageRank* give some interesting insights into this metric and the *GGBMC*. The four highest values for this metric have the following four competencies from the CSTA standards from 2011 [SCF⁺11]:

US1-14.4: The students are able to use standard output devices to successfully operate computer related technologies.

US1-14.3: The students are able to use standard input devices to successfully operate computer related technologies.

US1-14.2: The students are able to use standard output devices to successfully operate computers.

US1-14.1: The students are able to use standard input devices to successfully operate computers.

These competencies describe basic abilities, necessary for a lot of higher level competencies. This also reflects the purpose of the *PageRank* as it identifies basic competencies which important competencies depend on. Nonetheless, the impact of all four competencies on the whole *GGBMC* is fascinating.

After these four competencies some *intersector nodes* show the next highest values in the *PageRank*. The following *intersector node* presented with its combined competencies has the highest value of *intersector nodes*:

INT-005-DI:

- CH-14.0: The students are able to work with basic elements of the user interface (windows, menu, several opened programs) [Leh14].
- DE-15.0: The students are able to interact purposefully with information technology systems [Ges19].

Tab. 6.2: Comparison of the identified nodes by different centrality measures in the GGBMC (the colors highlight nodes which appear in different measures)

out-degree centrality	in-degree centrality	betweenness centrality	Page Rank
INT-014-DA	US1-3.0	INT-017-HF	US1-14.4
INT-011-DA	INT-009-HF	INT-016-HF	US1-14.3
INT-008-HF	INT-012-DA	INT-004-DA	US1-14.2
INT-009-HF	INT-005-DI	INT-014-DA	US1-14.1
INT-012-PA	INT-017-HF	INT-009-HF	INT-005-DI
INT-005-HF	US1-14.4	INT-019-DA	INT-017-HF
INT-011-HF	INT-18-DA	INT-020-DA	INT-016-HF
INT-019-PA	INT-20-DA	INT-18-DA	INT-18-DA
INT-002-DA	INT-13-PA	INT-011-DA	US1-3.0
INT-013-DA	INT-006-HF	INT-005-DA	INT-005-DA

This *intersector node* is again very basic and important for higher level competencies. None of the *intersector nodes* with high values in *PageRank* are *sinks*. This is of interest as both metrics help to identify basic competencies.

The first competency that is not from the CSTA standards from 2011 is on the thirteenth place after five competencies from CSTA 2011 and seven *intersector nodes* and comes from the Australian curriculum:

AU-1.0: The students are able to identify how common digital systems (hardware and software) are used to meet specific purposes [Aus13].

Again, this is a very basic competency and an obvious requirement for higher level competencies. These examples show that *PageRank* helps in the identification of basic competencies which are necessary for important higher level competencies.

Like in the results from the single curricula in some cases also different centrality measures of the *GGBMC* identify the same competencies. In Tab. 6.2, the ten nodes with the highest values in the corresponding centrality are presented and compared. The colors show if they are identified in more than one measure. Comparable to the results of the single curricula, in case of the *GGBMC*, the *out-degree centrality* identifies the lowest number of competencies which are also identified by other measures. Three competencies are shared with the *betweenness centrality* and one with *in-degree*. The *in-degree centrality* shares one identification with *out-degree*, four with *betweenness centrality*, and five with *PageRank*. *Betweenness centrality*

and *PageRank* have four nodes in common. So, the trend found in the analysis with the single curricula can not be seen here. Only the relation between *out-degree* and *betweenness centrality* can be assumed.

6.3.2 Intersector Nodes as Central Competencies

The *GGBMC* provides an additional way to identify central competencies, as the *intersector nodes* can also be seen as central. They combine competencies which occur at least in two different curricula. This means they are seen to be important for students by at least two curriculum-development groups. Some *intersector nodes* stand out because they have a high number of bundled competencies. The highest number with 10 bundled competencies shows the following *intersector node*, presented again with all its bundled competencies:

INT-005-PA:

- AU-10.0: The students are able to record design ideas using techniques including labelled drawings, lists and sequenced instructions [Aus13].
- US1-4.1: The students are able to use writing tools to illustrate thoughts in a step-by-step manner [SCF⁺11].
- US1-4.2: The students are able to use digital cameras to illustrate thoughts in a step-by-step manner [SCF⁺11].
- US1-4.3: The students are able to use drawing tools to illustrate thoughts in a step-by-step manner [SCF⁺11].
- US1-4.4: The students are able to use writing tools to illustrate ideas in a step-by-step manner [SCF⁺11].
- US1-4.5: The students are able to use digital cameras to illustrate ideas in a step-by-step manner [SCF⁺11].
- US1-4.6: The students are able to use drawing tools to illustrate ideas in a step-by-step manner [SCF⁺11].
- US1-4.7: The students are able to use writing tools to illustrate stories in a step-by-step manner [SCF⁺11].
- US1-4.8: The students are able to use digital cameras to illustrate stories in a step-by-step manner [SCF⁺11].
- US1-4.9: The students are able to use drawing tools to illustrate stories in a step-by-step manner [SCF⁺11].

It is visible that *INT-005-PA* contains competencies from two curricula. There are nine competencies from the CSTA standards from 2011 and one from the Australian

curriculum. The one competency from the AU is more general than all the others from the CSTA standards. While the CSTA standards differentiate between the technologies used to record ideas, this is not mentioned in the AU. Recording design ideas is also understood to be more general than recording simple ideas and often requires the illustration of thoughts as well as some kind of story telling. Therefore, the overall topic of all nine competencies from the CSTA standards are combined with the more general formulation of the Australian curriculum.

With six competencies, the following *intersector node* has the second highest number of bundled competencies:

INT-020-DA:

- AU-5.3: The students are able to organise ideas using information systems [Aus13].
- AU-5.4: The students are able to organise information using information systems [Aus13].
- GB-4.3: The students are able to use technology purposefully to organise digital content [Ber15].
- US2-5.2: The students are able to copy information using a computing device [CST17].
- US2-5.4: The students are able to retrieve information using a computing device [CST17].
- US2-5.6: The students are able to delete information using a computing device [CST17].

Here, again a generalization is recognizable. In this case the most general formulation derives from the model for the English curriculum referring to 'organise digital content'. The two competencies from the Australian curriculum use 'organise ideas' and 'organise information' which, in the context of information systems, could both be interpreted as 'digital content'. The CSTA standards from 2017 only use the term 'information' but define the actions done with it, so 'copy', 'retrieve', and 'delete', in more detail. All these actions can be summarized with the term 'organize'.

Also, the following *intersector node* combines six competencies:

INT-003-DA:

- AU-5.1: The students are able to create ideas using information systems [Aus13].
- AU-5.2: The students are able to create information using information systems [Aus13].

- CH-7.1: The students are able to design simple picture documents [Leh14].
- CH-7.2: The students are able to design simple text documents [Leh14].
- CH-7.3: The students are able to design simple sound documents [Leh14].
- GB-4.4: The students are able to use technology purposefully to create digital content [Ber15].

Again, six competencies from three curricula are combined which is already discussed in section 4.6.4. Because of this generalization *intersector nodes* with a very high number of combined competencies are generated. This process is correct, but that means that the number of bundled competencies is no indication for the importance of an *intersector node*, as they do not reflect the number of curricula considering these competencies as important. So, only those *intersector nodes* with a high number of competencies from different curricula are seen to be central. The next highest number of bundled competencies in an *intersector node* is four and can be measured in eleven of all generated *intersector nodes*. Five of them are a combination of competencies from four curricula and including the two nodes *INT-009-HF* and *INT-014-DA*. Both of them are discussed in the previous section as they also show the two highest *degree* values in the *GGBMC*. The importance of these bundled competencies is clear, as they appear in more or less the same formulation in four independent curricula. These examples show that it makes sense to consider *intersector nodes* with a high number of competencies from different curricula as central nodes.

6.4 Determination and Optimization of Learning Paths

6.4.1 Definitions and Research

Learning pathways or *learning paths* are no new concept in teaching. In science education this concept came up in 1991 and was later defined by Niedderer and Goldberg as follows [NG96, p. 2]

We describe learning pathways by describing a learning route of cognitive states, starting with prior conceptions and coming to intermediate conceptions during teaching.

They further describe that 'conceptions are seen as cognitive elements related to the special content domain' and understand the conceptions as 'current constructions' occurring in the students' minds [NG96]. So, students start learning at a given point and change these state during teaching.

The concept of *learning paths* is often used in e-learning research without defining it. Jih describes the role of *learning paths* as follows [Jih96, p. 1]:

Learning pathways also reveal the learning trails while learners traverse any interactive environment. Since learners have unique knowledge structures based upon their experiences and abilities, the ways that they choose to access, interact, and interrelate messages in interactive courseware also vary. Studies on pathways help us to explore and explain human behaviors during learning processes.

Here, *learning paths* are not understood as the stages during teaching but the actions learners take during an interaction process with an e-learning environment. By connecting knowledge units, Zhao and Wan model an e-learning course as a graph and describe *learning paths* as the way to get from one owned knowledge unit to a desired knowledge unit [CL06].

In the case of this dissertation a combination of the definition from Niedderer and Goldberg and the understanding from Zhao is used for *learning paths*. Competencies represent a form of stages or cognitive states in the learning process and take in the graph representations of the curricula the roles of knowledge units. Therefore, the following definition is used:

Definition 6.1 (Learning paths). A **learning path** in a graph representation G of a curriculum is a $[c_t, c_r]$ -**path** in G where c_t is the targeted competency (node) and c_r the already reached competency (node).

It has to be considered that in the graph representations of this approach dependency relations are used. This means that the directions of the edges in the graphs are pointing from higher level competencies to lower level competencies. That is why *learning paths* start at the target competency and end at a already reached competency.

As this definition of *learning paths* is based on the mathematical definition of a *path* it is defined between two nodes. This means a *learning path* can not include dependencies of more than one branch of a graph. Nodes representing *competencies* can *require* or *expand* more than one other competency. So, a *learning path* does not consider all the given dependencies. To represent the whole spectrum of dependency the additional construct *prerequisites graph* is defined as follows:

Definition 6.2 (Prerequisites graph). A **prerequisites graph** of a competency (node) c_t in a graph representation G of a curriculum is a subgraph of G including all paths from c_t to a set C_r of already reached competencies (nodes) of the graph. If C_r is empty all paths from c_t to sinks are considered.

A *prerequisites graph* can be used to gather information about all the dependencies of a competency. On basis of this, a teacher can decide which prerequisites are the most important ones and choose one or more *learning paths* to reach the targeted competency. Or the teacher uses given metric values to get suggestions for optimized *learning paths* as described in the following section.

6.4.2 Methods

There are different ways to optimize the *learning paths* for students. One way is presented by Zhao and describes the *shortest path* as the optimum for students as it theoretically consumes least time and effort. This requires weights on the edges which represent the difficulty of getting from one knowledge unit to another and are managed by teachers of the courses [CL06]. But saving time is only one aspect of an optimal *learning path*. It can also be considered that important or central nodes should be covered, or that the most basic competencies are included. As in this dissertation the relations are not weighted, three ways of learning path optimization are developed, based on the number of nodes and centrality measures.

- **Shortest Path:** the graph representations of the curricula do not include weights on their relations. Because of that, the *shortest paths* are determined by counting the number of visited nodes. This has the effect that the probability is higher to find several *shortest paths* for one current competency to a targeted competency.
- **Covering Central Nodes:** all nodes in the single graphs as well as in the GGBMC have calculated centrality values which indicate their importance for the curricula or the GGBMC. It can be argued that the nodes with the highest centrality values are important for the students and should therefore be covered by a *learning path*. For this reason, the second approach to determine *learning paths* is to select the path from a current competency to a target competency which includes the competencies with high centrality values as *learning path*.
- **Highest Overall Centrality:** the third approach also uses the centrality values of the nodes and sums them up to get the overall centrality of a path. This method can result in the same *learning paths* as the second method does. But if there is a path that includes several competencies with an average centrality, it could have a higher overall centrality. So, this approach favors *learning paths* with more competencies.

It has to be mentioned that it can happen that not all of these three methods only deliver one *learning path*. This is the case if the same values for the metrics occur in different competencies, if one competency is part of several paths and shows a high value for a metric, or if the sums of the values happen to be equal. So, the result of these calculations is a set of *learning paths* which can be empty if a *sink* is selected as *target competency*. With the help of these three methods *learning paths* in the single curricula and the GGBMC can be generated. In the following section examples of *learning paths* from single curricula and from the GGBMC are discussed.

6.4.3 Exemplary Use Cases and Reflection

In this section two *learning paths* are discussed as they are generated in the single curricula and in the *GGBMC*. Both examples are based on the assumption that no competency level has been reached so far and therefore no previous knowledge is present. This means that the starting nodes have to be *sinks*. The first one is a short example and is part of the CSTA standards from 2011. It is selected because it includes the node with the highest *in-degree* value in the *GGBMC*. Fig. 6.6 shows the *prerequisites graph* of the competency *US1-24.0* in the graph representation of the single curriculum. That means it includes no *intersector nodes*. The target node *US1-24.0*, representing the competency that should be reached, is a *source* and the starting nodes are the two *sinks* *US1-5.01* and *US1-14.1*. The *prerequisites graph* covers the following nodes:

1. **US1-24.0** The students are able to make a list of sub-problems to consider while addressing a larger problem [SCF⁺11].
2. **US1-5.2** The students are able to understand useful order, such as sorting students by birth date [SCF⁺11].
3. **US1-5.1** The students are able to understand how to arrange information [SCF⁺11].
4. **US1-3.0** The students are able to use technology resources to solve age-appropriate problems [SCF⁺11].
5. **US1-14.4** The students are able to use standard output devices to successfully operate computer related technologies [SCF⁺11].
6. **US1-14.3** The students are able to use standard input devices to successfully operate computer related technologies [SCF⁺11].
7. **US1-14.2** The students are able to use standard output devices to successfully operate computers [SCF⁺11].
8. **US1-14.1** The students are able to use standard input devices to successfully operate computers [SCF⁺11].

As Fig. 6.6 shows, in this *prerequisites graph* two *learning paths* to reachable *sinks* are possible. The figure of the graph is a screenshot from the *GECKO* platform presented in chapter 7. Yellow nodes represent *sinks*, orange ones are *sources*, and blue nodes are non special competencies. Depending on the selected method, one of the following two options is the result.

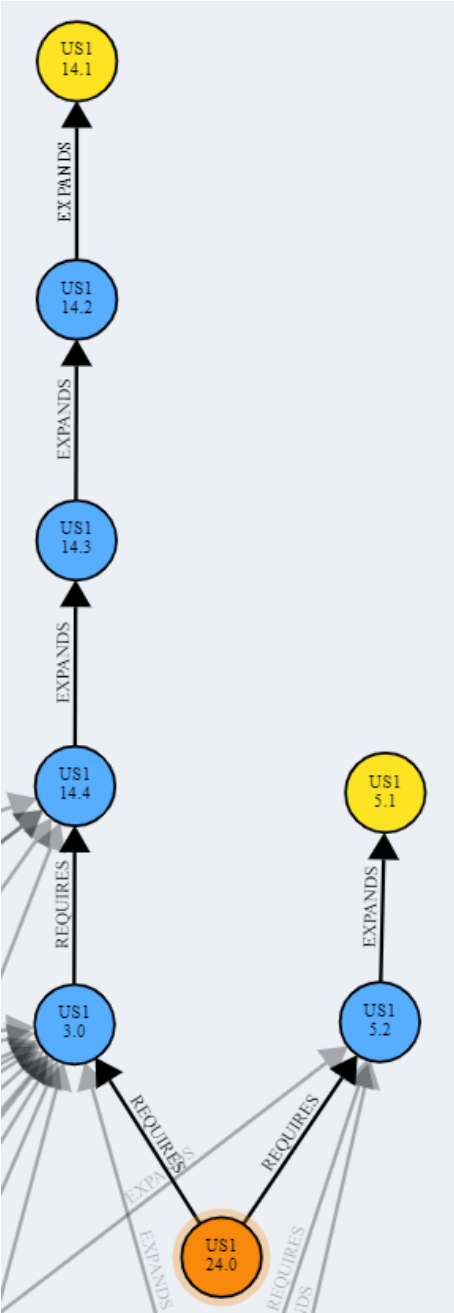


Fig. 6.6: Prerequisites graph for the node US1-24.0

1. **US1-24.0**
 US1-5.02
 US1-5.01

2. **US1-24.0**
 US1-3.00
 US1-14.4
 US1-14.3
 US1-14.2
 US1-14.1

Only the *shortest path* method results in *learning path* 1. The highest value of the *out-degree* is in both cases the same, so the *covering central nodes* method based on *out-degree* returns both paths. All other variants, *covering central nodes* based on *in-degree*, *betweenness centrality*, or *PageRank*, and *highest overall centrality* using the values from *out-degree*, *in-degree*, *betweenness centrality*, or *PageRank*, result in *learning path* number 2.

In the *GGBMC* some things changed for the same target competency *US1-24.00*, as it is part of the following *intersector node* and shows incoming dependencies:

INT-004-PA:

- US1-24.0: The students are able to make a list of sub-problems to consider while addressing a larger problem [SCF⁺11].
- US2-11.0: The students are able to decompose (break down) the steps needed to solve a problem into a precise sequence of instructions [CST17].

Additionally, the two competencies *US1-5.1* and *US1-5.2*, which are part of the *prerequisites graph*, are combined with one competency from the curriculum from Switzerland to the following *intersector node*:

INT-018-DA:

- CH-9.0: The students are able to sort things according to their own chosen properties so that they can find an object with a certain property more quickly [Leh14].
- US1-5.1: The students are able to understand how to arrange information [SCF⁺11].
- US1-5.2: The students are able to understand useful order, such as sorting students by birth date [SCF⁺11].

These differences to the single curriculum result in slightly different results for the determination of *learning paths*. Fig. 6.7 shows the *prerequisites graph* of the *intersector* and target node *INT-004-PA*, in which a third graph branch from this

node to a reachable *sink* can be seen. In this screenshot from the *GECKO* platform, again yellow nodes are *sinks*, blue ones represent non special nodes, and the magenta nodes are *intersector nodes*.

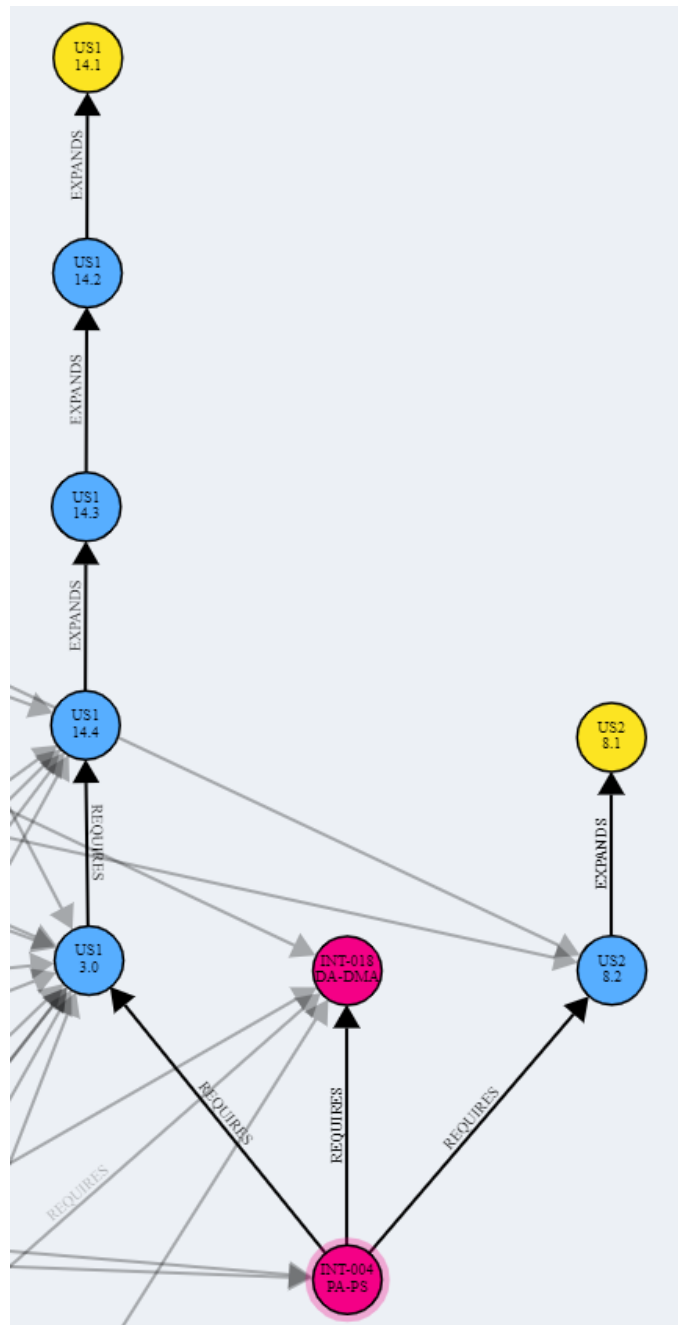


Fig. 6.7: Prerequisites graph for the node INT-004-PA

The combination of competency *US1-24.00* with competency *US2-11.0* from the CSTA standards from 2017 to an *intersector node* adds a relation to competency *US2-8.2* also coming from the CSTA standards from 2017.

US2-8.2 The students are able to model daily processes by creating algorithms to complete tasks [CST17].

US2-8.1 The students are able to model daily processes by following algorithms to complete tasks [CST17].

So, the *sink* *US2-8.1* can be added to the starting nodes. In the *GGBMC* the following three *learning paths* from *INT-004-PA* to reachable *sinks* are identified:

1. **INT-004-PA**
 INT-018-DA
2. **INT-004-PA**
 US1-3.00
 US1-14.4
 US1-14.3
 US1-14.2
 US1-14.1
3. **INT-004-PA**
 US2-8.2
 US2-8.1

The shortest path leads to the *intersector node* *INT-018-DA* which includes the two competencies which are also part of the shortest *learning path* in the single curriculum (*US1-5.02* and *US1-5.01*). Because it connects components of the *GGBMC*, the *intersector node* *INT-018-DA* has a very high *betweenness centrality* value. That is the reason why *learning path* 1 is also the result for *covering central nodes* and *highest overall centrality* based on *betweenness centrality*. The method *covering central nodes* based on *out-degree* returns all three possible *learning paths* as results. Using *in-degree*, or *PageRank* results in *learning path* number 3. The method *highest overall centrality* yields in all cases except the *betweenness centrality* the *learning path* 3 as result.

The results of the *learning path* determination in this example show that the combination to the *GGBMC* has impact on the metric values and therefore on the results for *learning paths*. This impact may be small but in more complex examples it gets more meaningful.

6.5 Conclusion

Analyzing the *Generic, Graph-Based Model for Competencies (GGBMC)* adds some interesting information to common curriculum analysis and to the results of the single curricula comparison. The values of the basic metric show differences to average values from the single curricula. In fact there are fewer *connected components* in the *GGBMC* which shows more connected categories. Also the number of *sinks* and *sources* are lower in the *GGBMC* than the average of the single curricula. This enables assumptions about the single curricula and their identified starting (*sinks*) or ending points (*sources*). It can be assumed that in some single curricula prerequisites for *sinks* are missing and that some curricula are going in some cases more into detail than others.

An analysis of the intersections between the single curricula, the *intersector nodes*, shows that more than a third of all competencies from the single curricula can be combined to *intersector nodes* following the formal process defined in section 4.6.4. The highest number of combined competencies can be found in the CSTA standards from 2017, whereas the GI standards show the lowest number.

The *centrality measures* also show differences between the *GGBMC* and the single curricula. All highest, normalized *degree* values, *overall degree*, *out-degree*, and *in-degree*, of the *GGBMC* are at the level of the lowest maximum values from the single curricula. Especially the highest, normalized *out-degree* values of the *GGBMC* are very low compared to the lowest maximum values. The trends of the identification of the same central competencies by different *centrality measures* found in the analysis of the single curricula, could on a first view not be verified in the *GGBMC*. Of the ten competencies with the highest values in all *measures*, *in-degree centrality*, *betweenness centrality*, and *PageRank* identify the same competencies in four to five cases. *Intersector nodes* are also seen as central nodes, as they appear in at least two curricula. But, it is not the number of combined competencies that is decisive but the number of different curricula present in one *intersector node*.

In addition to the already applied methods known from the comparison and analysis of the single curricula, the determination and optimization of *learning paths* are part of the analysis of the *GGBMC*. Therefore, the constructs *learning path* and *prerequisites graph* are introduced and defined. The determination of *learning paths* is based on the three methods *shortest path*, *covering central nodes*, and *highest overall centrality* which deliver a set of *learning paths*. For the optimization the *centrality measures* are used to identify different central competencies that can be covered by a *learning path*. So, if a *learning path* should cover basic competencies the methods are based on *PageRank*, if topics-connecting elements are seen to be important *betweenness centrality* is used, if competencies that are important to a lot of other competencies should be included *in-degree centrality* is selected, and if competencies that connect topics by depending on several competencies should be

part *out-degree centrality* is applied. The example shows that in simple cases often the same *learning paths* are suggested. In case of more complex examples the rule applies, that the more complex the *prerequisites graph* is, the more the results differ from the different methods.

Part III

IMPLEMENTATION

7. GRAPH-BASED ENVIRONMENT FOR COMPETENCY AND KNOWLEDGE ITEM ORGANIZATION

7.1 Introduction and Existing Research

On behalf of this dissertation different tools are developed to enable a semi-automated organization and analysis of curricula. All of them are based on the graph-based approach presented in the previous part. The first tool called *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*¹ is a web-based platform to display, evaluate and develop curricula for computer science related topics. The curricula are mapped to a graph database (neo4j) and are represented as directed dependency graphs. To improve administration, researchers and developers need a comfortable and fast way to upload new curricula or edit existing curricula in the graph database and represent them as graphs. The relations between the competencies of the curricula as well as the nodes representing the competencies can be evaluated by experts. For the purpose of a user friendly tool, the platform provides a comfortable, fast and motivating user interface for experts' evaluations. The results from these evaluations are stored and displayed within the tool. Researchers and developers with a valid account are able to retrieve these results in different statistical forms. Additionally, the tool provides teachers or curriculum developers the possibility to develop a new curriculum, following several criteria. So, teachers who are developing a curriculum for their class are able to filter the competencies they can reach, for example by a specific age-group, and receive suggestions which competencies the students should already have gained. They can add or delete suggested competencies to design their individual curriculum.

There exist different tools with a similar purpose like e.g. the online platform called *Software for Target-Oriented Personal Syllabus (STOPS)* from Auvinen, Paavola and Hartikainen presented in 2014 [APH14]. They describe it as 'a graph-based study planning and curriculum development tool'. Its purpose is to give university students the possibility to 'create personal study plans' and to enable teaching staff the administration of 'course content' and 'curriculum structure' [APH14]. In their graph-based models university courses are represented by intended learning outcomes and their prerequisite dependency relations. The dependency relations

¹ <https://gecko.aau.at>

can be 'mandatory' or 'supportive' whereby the latter represent elective courses. Teachers are able to use an editor to mark learning outcomes as 'mandatory' or 'supportive' and to change the sequence of the learning outcomes. In a graph visualization they get an overview of the course and the prerequisites. The teachers can also take a look at the learning outcomes from other courses. Students can browse study programmes, their lists of courses and learning outcomes and get information about the prerequisites. If they choose a course they get recommendations for supportive, elective courses. Additionally, students have the possibility to schedule their selected courses.

The *GECKO* platform developed during this dissertation offers school teachers of different levels as well as university lecturers the possibility to define their own curricula or *learning paths*. A large graph database of competencies from a broad source of different existing curricula, as it is described in section 4, builds the basis for this tool. Teachers get recommendations for competencies of a selected age group and can select starting competencies, where they want to start from, as well as target competencies, where they want to get with the students. On behalf of these information the system suggests a set of *learning paths* considering those prerequisites. In the following sections the tool and its development are described in more detail. Section 7.2 includes a detailed description of the tool and shows screenshots of the user interface. In section 7.2.2 various use cases are presented and the three different roles in the system are defined. Section 7.3 gives a basic overview of the systems architecture and section 7.4 discusses some challenges during development.

7.2 Tool Description

7.2.1 General User Interface

The web application starts with a log-in and registration interface. Also the password recovery can be found at this point. An account is necessary for all users to get access to the application. In the application the navigation is placed on the left hand side and differs for the different roles. All roles have the rights to access the graph representation and get a view on the single curricula as well as on the *GGBMC*. Fig. 7.1 shows the user interface and a part of the graph representation of the GI standards from Germany. In the navigation the list of graphs can be found. Here, all integrated single curricula are included as well as the *main graph* which represents the *GGBMC*. These graphs can be loaded separately or together in form of the *main graph*. On top of the graph the name (in this case the country code is used) and the number of nodes and edges are displayed. The colors in the graph shown in Fig. 7.1 give already some information about the graph as they present results from calculations described in section 5.4.4. Blue nodes are basic competencies, whereas yellow nodes represent *sinks* and orange nodes are *sources*.

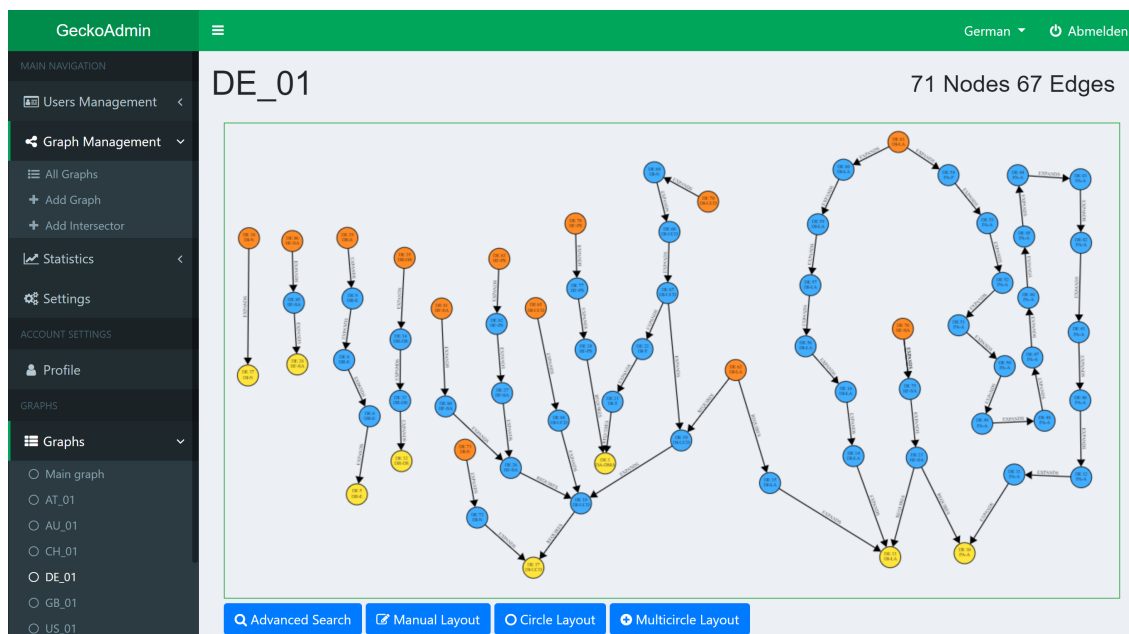


Fig. 7.1: Screenshot of the GECKO web platform

So, yellow nodes are good starting points for teaching. The presented graph follows no given layout form. The nodes are placed manually which is one possibility to structure the nodes. On the bottom of the graph four buttons can be found, one is to open a search and filter window, the other three enable a change of the layout from *manual* (default) to *circle* or *multicircle* layout and back to the *manual* layout. Further layout forms are part of future development. With the button 'Advanced Search' a field opens, where on the one hand the user can search for competencies by their *ID*, and on the other hand the competencies can be filtered for attributes like *country*, *code*, *level*, *minimum age*, *maximum age*, and *category*.

Each node shows its text when moving the mouse cursor over it. On click a node is highlighted and the *competency information bar* opens on the right hand side. Like the navigation bar, this bar offers different information and functionality for the different roles in the system. The general properties of the node like e.g. the *name*, the *ID*, the *category* (*strand*), the *subcategory* (*substrand*), the *text* (*description*), *level*, *minimum age*, and *maximum age* are visible for all roles. This bar can be seen in Fig. 7.2 again in an example competency of the GI standards from Germany.

The *competency information bar* also opens the possibility to show the *prerequisites graph* and *learning paths* to the selected node which can be seen in Fig. 7.3. *Prerequisites graphs* and *learning paths* in this version are calculated for single nodes and display their paths back to *sinks* which represent starting points. A user can decide which kind of *learning paths* should be shown. As described in section 6.4.2

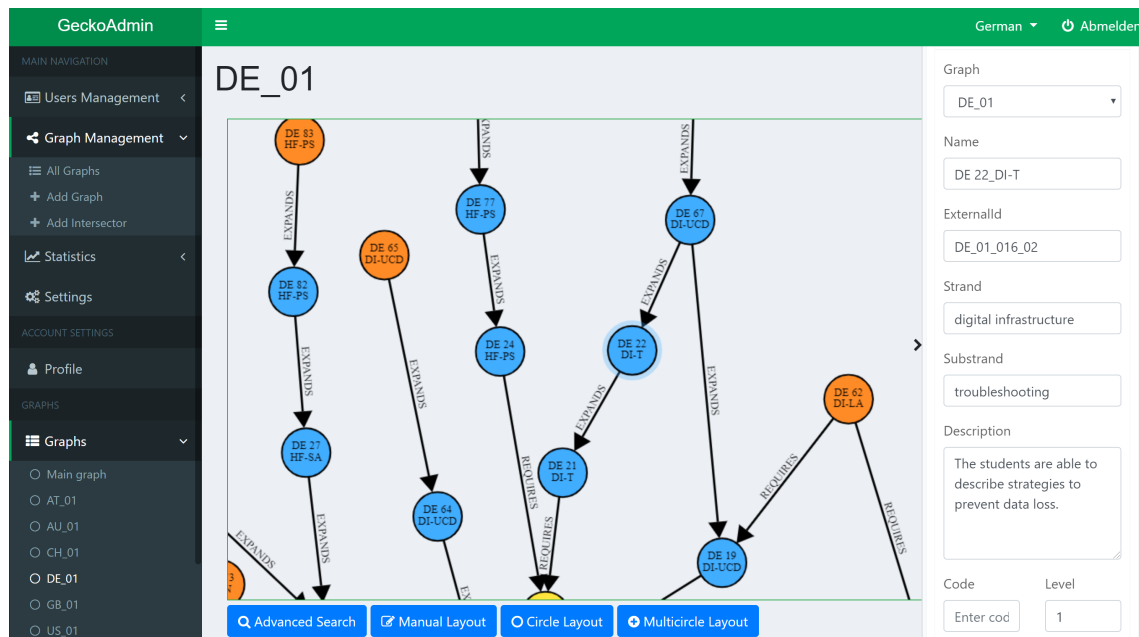


Fig. 7.2: Screenshot of the GECKO web platform showing the competency information bar

four possibilities are offered:

- None
- Shortest path
- Covering Central Nodes
- Highest Overall Centrality

If the option *None* is selected, all paths from the selected node, so the *prerequisites graph*, to *sinks* are highlighted. In the example presented in Fig. 7.3 the *shortest path* is selected. As this determination method does not need centrality values, the field below is not active. If one of the other two options is selected, the drop-down menu for 'Centrality Measure' can be opened. Here, the user can decide between:

- Betweenness centrality
- PageRank
- Node Degree Outgoing
- Node Degree Incoming

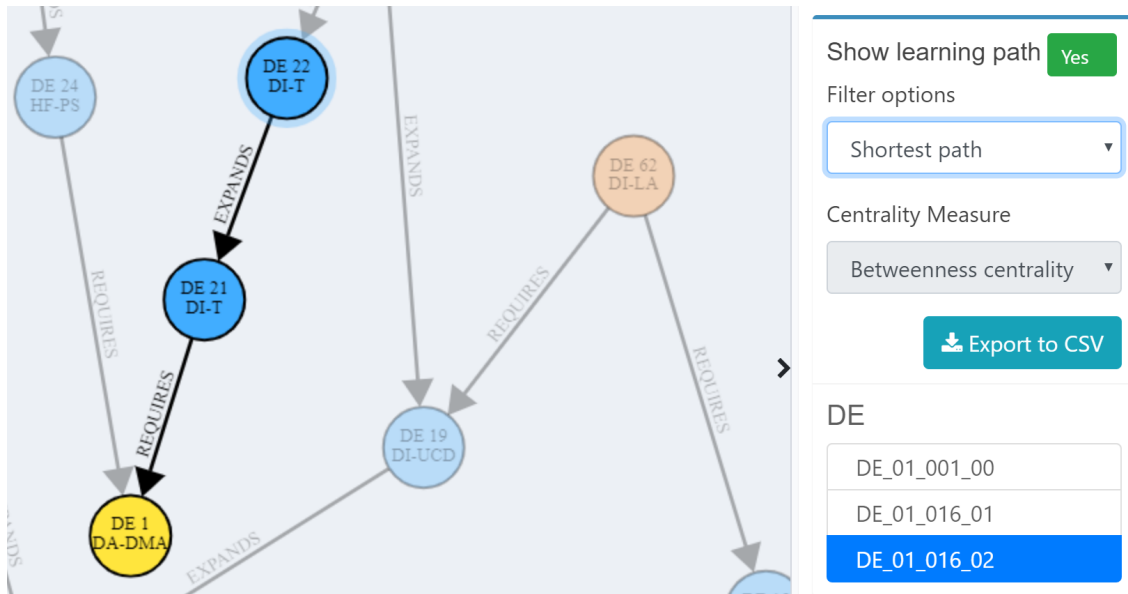


Fig. 7.3: The competency information bar showing a learning path for the selected node to a sink

Depending on the user's choice the *prerequisites graph*, respectively *learning paths*, is calculated and directly highlighted in the graph. For users like teachers or lecturers terms like *degree*, *betweenness centrality*, and *PageRank* may be unclear or unknown. Therefore, in the final version these expressions will be substituted by an explanation of which effects the corresponding measure has. Additionally, the list of visited competencies is shown in the *competency information bar* and can be downloaded as a CSV-file. If an *intersector node* is part of the *prerequisites graph*, respectively the *learning paths*, the visited competencies are grouped to the corresponding curricula.

With a click on a relation the *relations information bar* opens at the same position as the *competency information bar* pops up. It gives information about the two nodes the relation connects and about its properties. Fig. 7.4 shows a part of the *relation information bar*.

After this general view on the user interface, in the following section the roles and their use cases are discussed.

7.2.2 Use Cases and Roles

For the different use cases of the *GECKO* application the following three roles with different rights are provided by the system:

User: a standard user with the possibility to view graph representations and

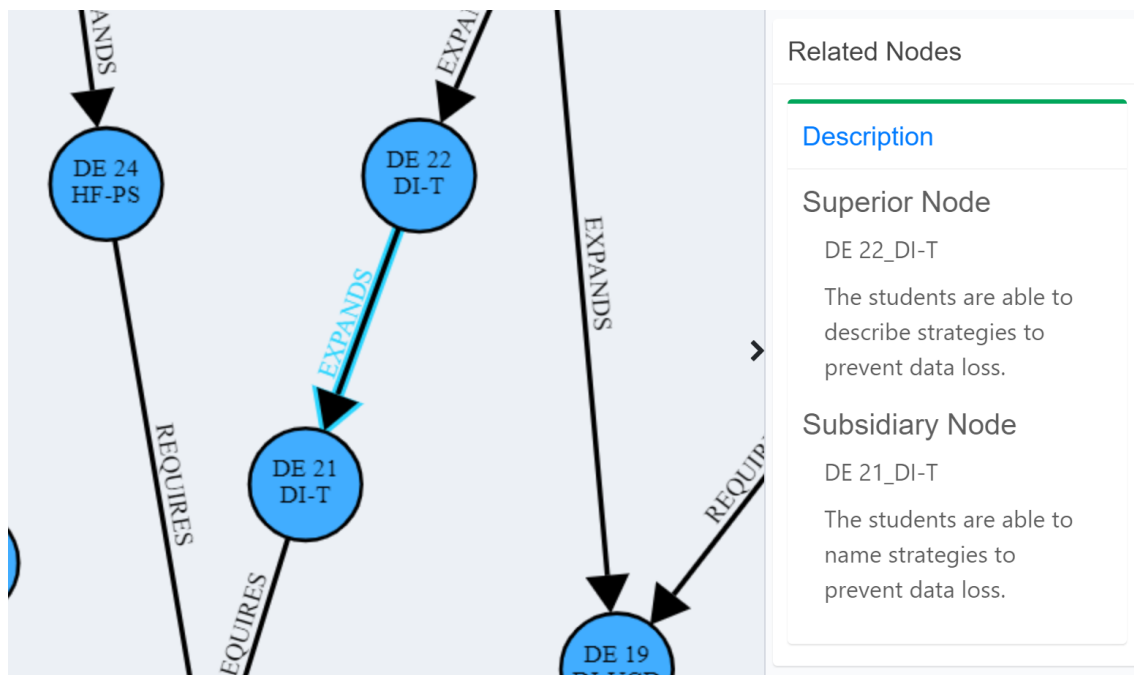


Fig. 7.4: A part of the relation information bar

build a own *learning path* or curriculum. The target group are teachers.

Expert: experts have to be invited or can be upgraded from standard users. They have the possibility to evaluate nodes and relations from the graph representations.

Administrator: besides user and graph administration this role has also the possibility to receive a statistical overview of the experts evaluations as well as the results for the calculated metrics.

The rights and use cases of these three roles are discussed in the following sections.

User

The additional features planned for this role are not yet part of the system but are part of future extensions. A standard user can register for free and only has to verify the e-mail address. In addition to general features, a user can develop and save own *learning paths* and curricula. The main target group for users are teachers and lecturers of computer science subjects or courses. They can determine *learning paths* for a certain competency which provide them with information about prerequisites and support them with additional ideas of which to teach. In their navigation field they can add, manage and change individual curricula. Teachers

and lecturers can use the search and filter form to find competencies which are of interest for them. Once they have found competencies, they can store them in one of their created curricula individually, as a whole *prerequisites graph*, or as a *learning path*. The system is also able to give suggestions for individual curricula. It only needs information such as the minimum and maximum age of the student group and the desired topic category.

Expert

As an expert must be an expert on the field of computer science education, this role is only assigned by administrators. People can be invited via e-mail or can press an 'Upgrade' button in their profile, if they are already registered as user. Experts have the same basic rights as standard users have and some additional possibilities to evaluate relations as well as nodes. On click on a node or a relation the *competency information bar*, respectively the *relation information bar*, opens. For a node the *importance* and the *complexity factor* can be rated. Additionally, the expert can give a comment if necessary. Fig. 7.5 shows the section of the *competency information bar* where experts can evaluate nodes. Experts can also evaluate the relations between the competencies. Clicking on the relation the *relation information bar* opens which for experts include a section 'Elements evaluation'. It provides the possibility to select if the relation is correct or not. If the relation is 'not correct', a field for comments opens. The expert should use this field to explain the decision. After the evaluation the relations are colored corresponding to their evaluation (red for 'not correct' and green for 'correct'). In the example of Fig. 7.6, the expert evaluated the relation as 'not correct'. The experts are informed about their evaluation progress in both *information bars*. A progress bar for competencies and for relations show them, how many of them they have already evaluated. For this dissertation the evaluation of the relations is relevant and some results are discussed in section 4.5.3.

Administrator

Besides the rights of users and experts and typical administrator tasks like user management, in the *GECKO* system the administrator is the only role with the possibility to create new content in form of additional curricula and to edit or delete existing graphs. New data can be entered online or uploaded in form of tables in CSV-files. The administrator can directly manage existing graphs by adding or deleting nodes or relations and editing properties. *Intersector nodes* can be generated from scratch or managed by adding or deleting combined nodes. The necessary data for these special nodes can also be uploaded in a similar form like the general graph data (tables in CSV-files). *GECKO* provides some statistical tools which can only be accessed by the administrator. It shows the progress and

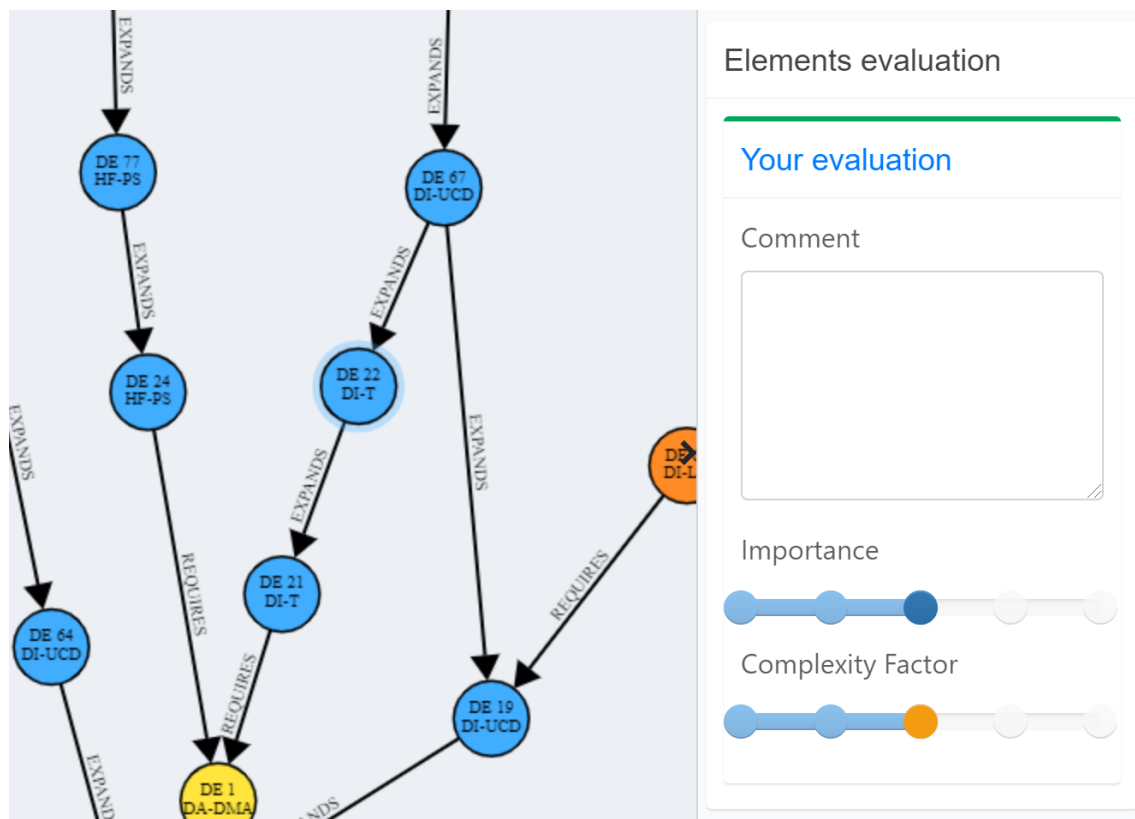


Fig. 7.5: The competency information bar with the possibility to evaluate a node

the results of the experts evaluation in lists as well as diagrams. Fig. 7.7 shows an example of the statistical overview of already evaluated elements. The blue sector signifies *not evaluated* elements, green stands for *agree* in the evaluation which means a relation is seen as correct, and red shows a *disagree* which means following the opinion of an expert the relation is incorrect. For all elements the evaluation results can be seen in a table. The top of this list is shown in Fig. 7.8. So, administrators can monitor the evaluation process and extract the results for further analysis. The administrator can also view the results of the calculations for the different metrics. In the navigation bar it is possible to select between *List of Node-Degree*, *Centrality Measures*, and *Other Metrics*. The *List of Node-Degree* contains a list of the *degree*, *in-degree*, and *out-degree* values of the nodes for a selected graph, including the main graph. In the *Centrality Measures* all results of *betweenness centrality* and *PageRank* can be found again in form of lists including the values for each node. Calculations for whole graphs like the *number of cycles*, the *number of connected components*, the *density*, the *number of sinks*, and the *number of sources* are presented altogether under *Other Metrics*.

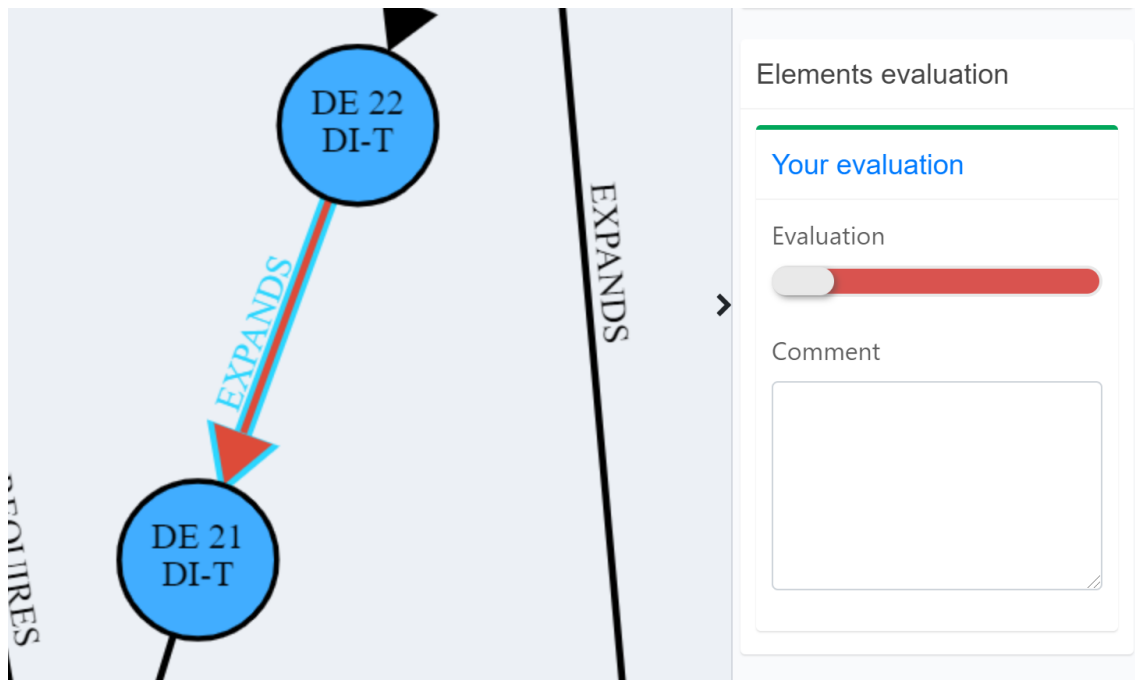


Fig. 7.6: The competency information bar with the possibility to evaluate a relation

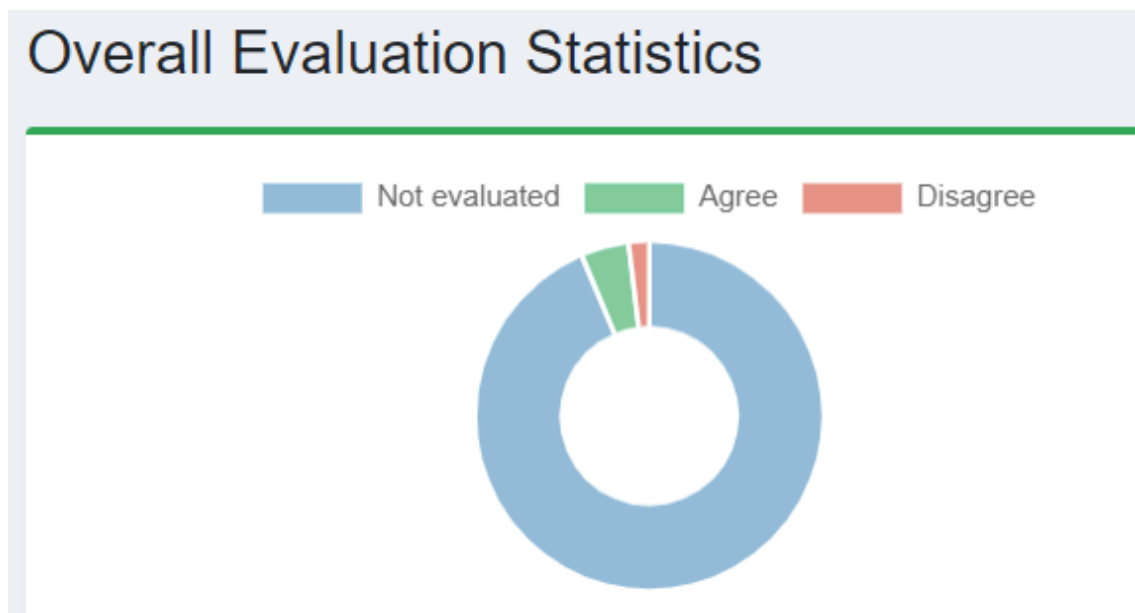


Fig. 7.7: Statistical overview of the number of evaluated elements and some results






Element name ▲	User ▲	Type ▲	Evaluation ▲
AT 100_DR-TD - AT 101_DR-TD		link	Agree
AT 101_DR-TD - AT 103_DR-TD		link	Agree
AT 109_HF-CC - AT 110_HF-CC		link	Agree
AT 10_HF-PS - AT 11_HF-PS		link	Agree
AT 110_HF-CC - AT 111_HF-CC		link	Agree

Fig. 7.8: A part of the overview of the evaluation results

7.3 Architecture

7.3.1 Technological Overview

The technological architecture of the *GECKO* application consists of several parts which can be seen in Fig. 7.9. A *neo4j* graph database is used to store the graph representations of the curricula. With the help of the RESTful Spring Service² *GeckoAPI* a connection to this graph database is established and data retrieval and queries are enabled. The *GeckoAPI* is called from the server side of the Website. Front and back-end of the systems are controlled by the *GeckoWeb* which connects to the *GeckoAPI* over RESTful. It provides a client-server architecture and is implemented with the PHP framework Laravel. To store user data, suggestions, evaluation, and basic graph data a MySQL database is included. The *GeckoAPI* and the *GeckoWeb* can retrieve data from this database using TCP/IP [Moe20] (see the development documentations of the project [TD17] and [Moe20] for more information about implementation and used technologies).

7.3.2 System Structure

An overview of the *system structure* in form of an entity relationship diagram can be found in Fig. 7.10. It contains the three different user roles *user*, *expert*, and *administrator* as entities. All users have the same three attributes: an identification number (ID), an e-mail address, and a password. A *user* can query a *learning path* and store it in the profile. As attributes the *learning path* has the *number of nodes* which is the number of covered nodes in the *learning path*, and the *overall centrality* which is a set of sums of the different centrality values of the nodes on

² Representational State Transfer (REST) is a popular technology to realize web services. A RESTful API follows the conventions of REST. Spring is the name of a Java EE framework.

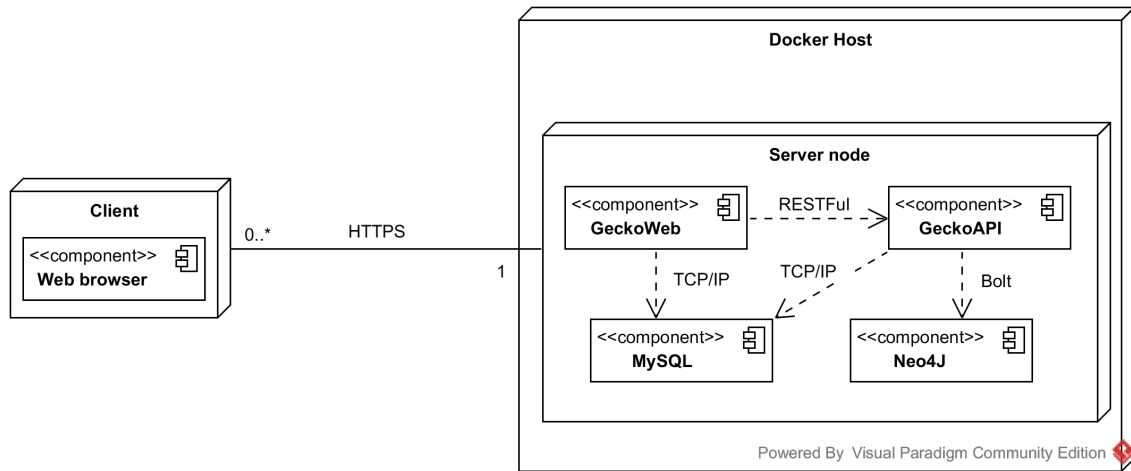


Fig. 7.9: The architecture of the GECKO system [TD17]

the *learning path*. When a *learning path* is stored by a user it gets a *title* and a *timestamp*. A *learning path* consists of *intersectors* and *competencies*. An *expert* makes evaluations for *competencies* (*competency expert evaluation*) and for *relations* (*relation expert evaluation*). Both evaluations have a *timestamp*, a *importance*, an *estimated duration*, a *status*: *not evaluated*, *OK*, *notOK*, a *suggestions (if notOK)*, and a *complexity factor*. The results are stored for each evaluated element.

An *administrator* organizes the three major elements *curriculum*, *competency*, and *intersector*. These elements represent the core entities which are stored in the graph database and are therefore highlighted in blue color. The relations between these core elements which are *expands*, *requires*, *is part of*, and *is bundled in*, are also part of the graph database and highlighted in dark red. A *curriculum* has the attributes *name*, *organization*, and *country*. Each *competency* is part of one *curriculum*. A *competency* can be related with another *competency* over an *expands* or a *requires* relation. Both relations have several attributes and the attribute *added*, which represents if a relation is added or given by the original curriculum, is the only attribute which is not evaluated or influenced by *experts*. The attributes *importance*, *estimated duration*, *status*: *not evaluated*, *OK*, *notOK*, *suggestions (if notOK)*, and *complexity factor* are dependent on the opinions of the *experts* and therefore derived attributes. Besides already explained attributes like e.g. *ID*, *text*, *level*, *minimum age*, *maximum age*, or *country*, *competencies* have some new attributes which are necessary for the development. Examples for new attributes are

hasSubs: a boolean that gives information if a competency has subcompetencies or not

bundled: is by default NULL and set to TRUE if a competency is included

in an *intersector node*

Competencies also have the multi-valued attribute *centrality measures* which contains all centrality values. The attributes *importance*, *complexity factor*, *category (strand)*, and *subcategory (substrand)* can be evaluated by experts during the *competency expert evaluation*. For translation aspects *competencies* are related to *languages* including the *name* and the translated *text*. A *competency* can be part of exactly one *intersector* which is realized with the relation *is bundled in*. In the graph the *competencies* can be connected to *intersector nodes* over *expands* and *requires* relations and vice versa. This is why in the entity relationship model shown in 7.10 these *expands* and *requires* relations have no directions. These relations are different to the relations between *competencies* as they base on *competencies* combined to *intersector node* and their relations. This also effects their attributes *importance*, *estimated duration*, and *complexity factor* which are calculated using the evaluated results from the combined *competencies*. An *intersector* has an *ID*, a *text* which is also extracted from the included *competencies*, a *minimum age*, a *maximum age*, a *category*, a *subcategory*, and the *countries* of all included *competencies*. *Intersectors* can *require* or *expand* other *intersectors* depending on their combined *competencies*. These relations have the same attributes as relations between *intersectors* and *competencies*.

7.4 Development

During the development process a lot decisions had to be made to provide planned features. In this section some extracts of the implementation done for the project are discussed which can be seen to be representative examples.

7.4.1 Semi-Automated Intersector Node Generation

The automation of the generation of *intersector nodes* was a challenge in design as well as practical perspective. The idea behind *intersector nodes*, as it is already discussed in section 4.6.4, is it to combine similar *competencies* to one element to connect different curricula with each other. That means, all the relations coming into and going out of a *competency* have to be transferred to the corresponding *intersector node*, as it substitutes the *competency*. This opens the question, how to deal with the substituted *competency*. Should it be deleted, overwritten or kept in the background? Each solution has advantages like e.g. increase of clarity or preservation of additional information but also disadvantages like e.g. increase of complexity or information loss. The solution found for this project is to keep the *competency* in the graph database, but set a flag in form of an attribute which marks it as being *bundled*. This attribute is used in queries to exclude those *competencies*

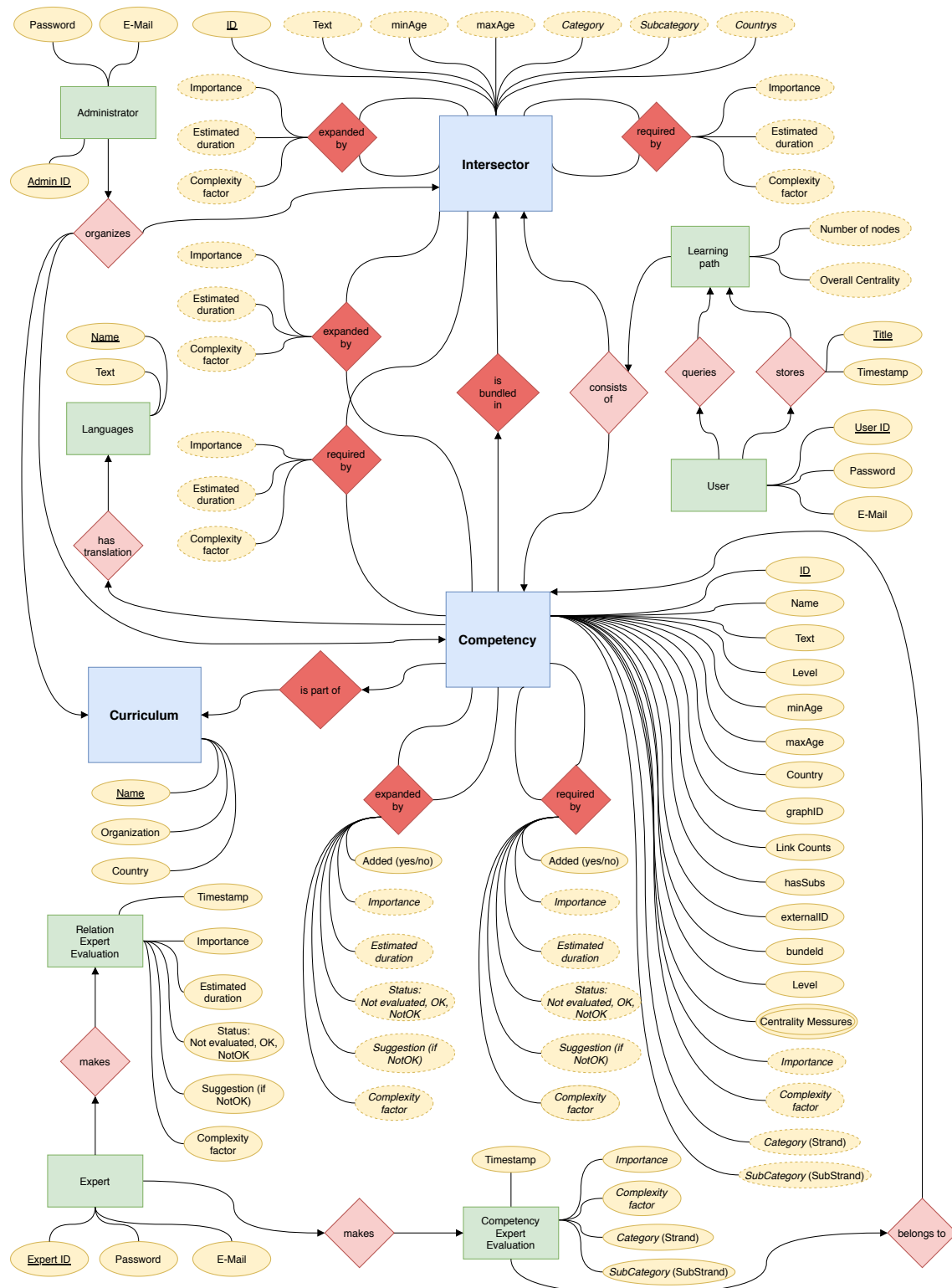


Fig. 7.10: The Entity Relationship Diagram for the GECKO system [Moe20]

```

1 MATCH (cu:Curriculum) WHERE ID(cu) = {id}
2 CALL algo.pageRank.stream('Competency', 'expands', {direction:'out',
  iterations:20})
3 YIELD nodeId, score
4 MATCH (c:Competency)-[:IS_PART_OF]->(cu)
5     WHERE id(c) = nodeId
6     RETURN c as competency, score as centrality
7     ORDER BY id(c) DESC

```

Listing 7.1: Query for the PageRank in single curricula for the *expands* relations [Moe20]

which are combined to an *intersector node* from calculations. So, the information included in the combined competencies are preserved and accessible. One reason for this solution is that single curricula are meant to stay in the system and users should be able to take a look at them as well as at the main graph. As *intersector nodes* are not existent in the single curricula, a deletion of the competencies would invalidate the underlying curriculum.

As a technical solution the node type *intersector* and the relation type *is bundled in* are added to the graph database. All competencies which are combined to an *intersector node* are connected over a *is bundled in* relation to the corresponding *intersector node*. At the upload or generation of an *intersector node* the system automatically copies all relations from the combined competencies to the *intersector node* and sets the mentioned attribute *bundled in* in the relevant competencies to TRUE. This improves performance when graph database is queried.

For the graph visualization in the *GECKO* platform this decision has the effect that in the *main graph (GGBMC)* the combined competencies and their relations are no longer visible. Instead the *intersector nodes* are shown and also connected to those competencies or other *intersector nodes* which are connected to the combined competencies [Moe20].

7.4.2 Queries for Centrality Measures

Essential parts of the system are the *centrality measure calculations* which are all done using queries for the graph database *neo4j*. These calculations are very well documented for *neo4j* in several sources like e.g. by Needham and Hodler in the book 'Graph Algorithms' [NH19]. Based on documented examples the queries for the calculations are adapted for the needs of this project. For the centrality measure *PageRank* of the *expands* relations in the single curricula the *cypher* query is shown in Listing 7.1.

Line 1 ensures that only competencies from one curriculum are considered. In line 2 with *CALL algo.pageRank.stream* the *PageRank* procedure provided in *neo4j*

```

1 MATCH (cu:Curriculum) WHERE ID(cu) = {id}
2 CALL algo.pageRank.stream(
3     MATCH (c:Competence) RETURN id(c) AS id ,
4     MATCH (u1:Competence) -[:REQUIRES|:EXPANDS]->(u2:Competence)
5         RETURN id(u1) AS source , id(u2) AS target ,
6     {iterations:20, graph:'cypher'})
7 YIELD nodeId , score
8 MATCH (c:Competence) -[:IS_PART_OF]->(cu)
9     WHERE id(c) = nodeId
10    RETURN c as competence , score as centrality
11    ORDER BY id(c) DESC

```

Listing 7.2: Query for the PageRank in single curricula [Moe20]

is started. As parameter it gets the node types (in the example 'Competency') and relation types (in the example 'expands') that should be considered. The additional parameter *direction* defines the direction of the relations which the algorithm should follow, and the parameter *iterations* changes the number of iterations. With *YIELD* the fields, which have to be returned, are selected, which in this case are the node number (nodeID) and the *PageRank* value (score). The *MATCH* and the *WHERE* clause define the elements to search for which are *competencies* with a *is part of* relation to the *curriculum* defined before. Here, the possibility to give the reading direction in *cypher* queries is visible. In line 4, after the relation type *is part of*, an arrow ($->$) to (*cu*) can be recognized. This symbols define the direction and can also be used in the other direction on the other side of the relation type declaration. If no arrow is given both directions are considered.

This first example is a simple variant of the algorithm which considers only one special type of relations. For the metrics calculated in this project both types of relations between competencies, *expands* and *requires*, are relevant. So, the algorithm has to be extended as Listing 7.2 shows. The only changes to the previous variant are the parameters the *CALL* clause in line 2 gets. To include both relation types an additional *cypher* statement is necessary, which is called a *cypher projection*, to load or project parts of the graph. For a *cypher projection* the *CALL* must include *graph:'cypher'* as it can be seen in line 6. The changes start at line 3 with a *MATCH* clause for all elements of the type *Competence*. In line 4 the second *MATCH* clause finds any competency that is related with other competencies over *expands* or *requires* relations. The results are returned as *source* and *target* nodes which replace the type of the relations to be considered.

For calculations in the *GGBMC* the algorithm even gets more complex as it must consider *intersector nodes* besides the competencies. Listing 7.3 shows this variant of the algorithm. In line 2 the *MATCH* clause identifies all *intersector nodes* and the results are combined by a *UNION* with the results from the second *MATCH*

```

1 CALL algo.pageRank.stream(
2     MATCH (i:Intersector) RETURN id(i) AS id
3     UNION MATCH (c:Competence)
4         RETURN id(c) AS id ,
5     MATCH (u1)-[:REQUIRES|:EXPANDS]->(u2)
6         WHERE u2.bundled IS NULL
7         RETURN id(u1) AS source , id(u2) AS target ,
8     {iterations:20,graph:'cypher'})
9 YIELD nodeId , score
10 MATCH (c)-[r:REQUIRES|:EXPANDS]->(c2)
11     WHERE c.id=nodeId
12     SET r.prAll=-score , c.prAll=score

```

Listing 7.3: Query for the PageRank including intersector nodes [Moe20]

clause finding all *competencies*. This again defines the node types to search for. The relation types are determined by the next *MATCH* clause beginning in line 5. In this statement elements, not only *competencies*, are found that are connected over *expands* or *requires* relations. Only those elements whose *bundled* attribute is NULL are considered (see line 6). Again results are given as *source* and *target* nodes. The last part of the query also differs to the previous variant. It does not deliver a list of competencies and centrality values, but sets the attributes *prAll* of the relation to the negative *PageRank* value and from the node to the true *PageRank* value (see line 12). So, centrality values are stored in node attributes as well as in relation attributes [Moe20]. The reason for this is a preparation for the *learning path* determination which is discussed in the subsequent development example.

7.4.3 Learning Paths Determination

As already mentioned in section 6.4, there are different ways to determine and optimize *learning paths*. In this version of the application a *learning path* is a path from a selected node, *competency* or *intersector node*, to a node without outgoing relation, so to a *sink* which can also be a *competency* or *intersector node*. The different options (see section 7.2) make different calculations necessary.

- **None:** to show the *prerequisites graph* all possible paths from the selected node to *sinks* are calculated and displayed.
- **Shortest path:** for calculation of the shortest path *neo4j* [NH19] uses the algorithm from Dijkstra [Dij59]. Each edge gets the length of 1. Some edges have to be excluded from the calculation and therefore get the maximum integer value. In case of single curricula these are the edges to *intersector nodes*, in case of the *GGBMC* these are edges to *bundled competencies*. Listing 7.4

```

1 MATCH (start), (end{outgoingLinksCount:0})
2     WHERE start.id={sourceId}
3 CALL apoc.algo.dijkstra(start, end, 'REQUIRES>|EXPANDS>', 'length')
4 YIELD path, weight
5     WHERE weight < 2147483647
6     RETURN nodes(path) as nodes, weight

```

Listing 7.4: Query for shortest path determination [Moe20]

shows the calculations for this option in *cypher*. The *start* is the selected node and *end*, as it can be seen, returns all nodes with outgoing link count of 0. In line 3 the procedure for the Dijkstra algorithm is called over outgoing *expands* or *requires* relations. With *length* the field to store the algorithm values is given. The *WHERE* statement in line 5 restricts the results to those paths with weights under the maximum integer value. The *RETURN* statement delivers all found nodes and the calculated weights. For this algorithm all possible paths have to be followed to find the one with the lowest weight.

- **Covering Central Nodes:** as the nodes have centrality values, these values can be used to determine paths covering nodes with a high centrality. So, all paths are calculated as described and the highest centrality value of included nodes is determined. All paths including this node are considered as results.
- **Highest Overall Centrality:** to calculate the paths with the highest overall centrality a trick is used because there is no algorithm for longest paths in *neo4j*. For that the centrality values of all nodes in a path have to be summed up and the paths with the highest sums are considered. At first the edges are assigned a negative centrality value (see previous example). With them a 'negative' Dijkstra algorithm is performed which sums up the negative weights and results in the paths with the highest negative value. The query for that can be seen in Listing 7.5 which looks very similar to the shortest path algorithm. The only difference is that only paths with a weight less than 0 are returned which can be seen in line 5. For the weights the name of the field, that contains the centrality values, are provided (see line 3 'degree') [Moe20].

7.5 Conclusion

This chapter provided some information about the *Graph-based Environment for Competency and Knowledge Item Organization (GECKO)* project and was developed as part of this dissertation. The main aims are to support the organization of

```

1 MATCH (start), (end{outgoingLinksCount:0})
2     WHERE start.id={sourceId}
3 CALL apoc.algo.dijkstra(start, end, 'REQUIRES>|EXPANDS>', {degree})
4 YIELD path, weight
5     WHERE weight<=0
6     RETURN nodes(path) as nodes, weight

```

Listing 7.5: Query for learning paths with high overall centrality [Moe20]

the elements from the curricula, and the calculations for analysis, as well as to enable the automated visualization of the graph representations, the online evaluation of competencies and relations, and the generation of individual learning paths or curricula. Several screenshots of the platform give insights of the user interface and provided features. For instance is described how learning paths can be calculated and retrieved. The different roles *user*, *expert*, and *administrator* and their rights, tasks and possibilities are discussed. Further, the software components as well as the structure of the system are briefly presented. To give also some insights into the development with *neo4j* some critical queries are described.

The application offers already a lot features but will be extended in future work. To simplify the evaluation process *experts* should get suggestions for additional relations if they think a relation is not correct. The competencies and relations should contain additional attributes which can be evaluated by *experts*. The calculation of *learning paths* should not be limited to a single node selection. The features for the *users* role has to be added to the system. Besides already described possibilities, *users* should be able to select a *reached* node, representing their students level, and a *target* node which they intend to reach. For the generation of their own curriculum *users* should be able to select several nodes press an *add* button, with the option to include prerequisites or not, and the system adds them so a also selected individual curriculum. So, the *users* are supported to generate their own curricula. A further example of an extension can be to add the role of *students* who can take a look which level in a given learning path they have reached and which comes next.

8. (SEMI-)AUTOMATED IDENTIFICATION OF COMPETENCIES AND RELATIONS

8.1 Introduction and Existing Research

As mentioned previously, different support tools are developed in the course of this work. The online platform called *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*¹ is described in the previous chapter and enables the organization of the graph representations of the curricula, the evaluation by experts, the determination of learning paths, and calculations for several metrics. To support the generation of the graph representation tools using techniques from *text analysis* and *natural language processing (NLP)* are developed. The main idea is to find similar terms in the competency formulations and to suggest, with the help of these similarities, relations between competencies, a convenient category, and the combination of competencies to *intersector nodes*.

Similar ideas are used in literature considering academic courses and undergraduate degree programs. For instance Badawy, El-Aziz, and Hefny [BEH16] analyze higher education textbooks on the basis of the included learning objectives. Their main aim is it to identify important chapters in textbooks according to intended learning outcomes of a curriculum. In their paper, they describe the steps taken during their research as follows [BEH16, p. 3]:

1. **Data Preparation:** in a first step the learning objectives of the textbooks and the intended learning outcomes from the curricula are collected.
2. **Getting Synonyms of Intended Learning Outcomes:** with the help of *Wordnet*² synonyms for the learning outcomes are determined.
3. **Data Processing:** in this step several smaller actions are included with the intention of preparing data for automated analysis. These steps are well known in the context of NLP:
 - (a) Tokenization: components of a text are separated into smaller tokens like sentences or words.

¹ <https://gecko.aau.at>

² <https://wordnet.princeton.edu/>

- (b) Stop words removal: very often occurring words like *a*, *and*, *about*, *to* are removed from the texts.
- (c) Stemming: words are changed to a standard form which can be their base or root form.

4. **Analysis of Learning Objectives:** the frequency of the occurring words from the intended learning outcomes in the learning objectives are measured and summarized for the chapters. With that the chapters get weights, whereas a high weight assumes a more important chapter for the intended learning outcome.

A more complex approach based on statistical methods comes from Sekiya, Matsuda, and Yamaguchi [SMY10] and focuses on topics and relations of a curriculum. They generate a map of a curriculum to analyze the structure and to get holistic understanding. A method called *latent Dirichlet allocation (LDA)* is used to extract topics from the syllabi of a curriculum and to calculate their relations. This method models the items of a text as a 'finite mixture over an underlying set of topics' and each topic as an 'infinite mixture over an underlying set of topic probabilities' [BNJ03, p. 1]. So, following Sekiya, Matsuda, and Yamaguchi 'LDA estimates a set of topics, where each topic is characterized by a distribution over words' [SMY10, p. 49]. The results from LDA are used as coordinates to generate maps of the syllabi.

The tools developed for this dissertation intend to analyze the competency formulations on a basic linguistic value and to provide suggestions about possible relations, categorization, or combination. Similar steps as described by Badawy, El-Aziz, and Hefny [BEH16] are processed during the preparation of the data, as they represent basic steps necessary for automated analysis [MRS08]. The tools differ in preprocessing as for this approach own synonym lists are developed and lemmatizing is applied instead of stemming. Relations are generated with the help of basic similarity measures instead of probabilistic models. Also different libraries are used. For preparation, the Python libraries *Natural Language Toolkit (NLTK)*³ and *spaCy*⁴ for NLP are used. Preprocessing, calculations, and similarity measures are done with the programming language *Python* ⁵. The library *gensim*⁶ is used in terms of similarity calculation.

In the following sections, the basic steps for text analysis and similarity measures used for this project are described (see section 8.2). The results from the text analysis of the curricula are presented in section 8.3. To improve the approaches, based on linguistic features, synonyms are collected and replaced. This process is

³ <https://www.nltk.org/>

⁴ <https://spacy.io/>

⁵ <https://www.python.org/>

⁶ <https://radimrehurek.com/gensim/>

described in section 8.4. In section 8.5 a possible approach to suggest relations between competencies is discussed including some results from before and after the synonym replacement. Two different approaches to automatically categorize the competencies into *computer science* and *digital literacy*, or into the five categories for computer science education are presented in section 8.6 including a discussion of the results. Section 8.7 describes an approach to find candidates for the combination into *intersector nodes* among the competencies (see section 4.6.4).

8.2 Text Analysis-Based Approach

8.2.1 Basic Steps

The basic preprocessing of a text corpus before automated analysis is in most cases straight forward and includes several steps. It aims, to enable an automated analysis and thus to bring the data in a form that can be processed easily. Literature suggests to do so by *indexing* which is the process of marking the occurrence of elements like e.g. terms with numbers [MRS08]. This results in an *incidence matrix* with the elements and the documents. The so called *inverted indexing* lists the elements in a *dictionary* including pointers to documents the elements appear in. Manning, Raghavan, and Schütze summarize the major preparation steps for *inverted indexing* as follows [MRS08, p. 19-34]:

1. **Collect the documents to be indexed:**

- **Obtaining the character sequence in a document:** in the case of this dissertation the documents are the lists of competencies included in the selected curricula. Each list from one curriculum is collected in CSV-documents which makes an automated import for an analysis easier.
- **Choosing a document unit:** the unit of interest is the term, or also compound terms, as they include information about a topic or the cognitive level.

2. **Tokenize the text:** depending on a given document unit, texts are divided into pieces which are called *tokens*. So, e.g. tokens of a paragraph are sentences separated by a point and a space, tokens of a sentence are terms separated by a space.

3. **Do linguistic preprocessing of tokens:**

- **Dropping common terms (stop words):** very common words, which have no value for the text analysis, are called *stop words* and are removed from the given corpus. In most cases, also for this dissertation, these words are collected in a list and excluded from further processing.

- **Stemming and lemmatization:** as in the documents words with the same derivation can appear in different forms like e.g. singular or plural, or different tenses, it is necessary to bring them to a basic form. This can be done by *stemming* or *lemmatization*. *Stemming* works very straight forward and chops the ends of given words. In the majority of cases this can lead to good results. With this process derivation affixes are removed. *Lemmatization* aims to return base or dictionary forms of words which is called *lemma*, using dictionaries or morphological analysis.

These steps are also part of the preparation for this project. Additionally, *part of speech tagging (POS)* is performed which classifies found words into nouns, verbs, etc. In several cases also more detailed information is delivered as classes like '*Noun, proper*' (*NNP*) or '*Verb, gerund*' (*VBG*) are used [MRS08, p. 40].

For a first analysis simple measures like the *term frequency*, *inverse document frequency*, and *tf-idf* are determined and discussed. The *term frequency* simply represents 'the number of occurrences of a term t in document d ' [MRS08, p. 117] and is denoted by $tf_{t,d}$. As the *term frequency* suffers the problem of considering all terms equally important, other methods are commonly applied. One of them is the *inverse document frequency (idf)* and it is based on the *document frequency* df_t which is 'the number of documents in the collection that contain a term t ' [MRS08, p. 118]. The *inverse document frequency* of a term t is defined as follows [MRS08, p. 118]:

Definition 8.1 (Inverse Document Frequency (idf)).

$$idf_t = \log \frac{N}{df_t}$$

Where,

N is the number of documents in the collection, and

df_t is the document frequency of t .

As far as the *inverse document frequency* is concerned, terms with a high frequency tend to have low *idf* values, rare terms often show high *idf* values. A third often used measure is the *tf-idf* value which represents a combination of *term frequency* and *inverse document frequency*. As the *term frequency* and the *inverse document frequency* do, the *tf-idf* adds a weight to a term. It is calculated for a term t and a document d as follows [MRS08, p. 119]:

Definition 8.2 (tf-idf).

$$tf - idf_{t,d} = tf_{t,d} \times idf_t$$

```

1 def tokenizeTagLemmatize(text):
2     tagged = tags(text)
3     lemmatizer = WordNetLemmatizer()
4     result = []
5     for t in tagged:
6         tag = getTagCategory(t[1])
7         if (tag != "O"):
8             result.append(lemmatizer.lemmatize(t[0], pos=tag) + "-" +
9                             getTagCategory(t[1]))
10    return(result)

```

Listing 8.1: Python function used for tokenization, tagging and lemmatizing [PK18].

Following Manning, Raghavan, and Schütze the *tf-idf* has three characteristics [MRS08, p. 119]:

1. it is highest when t occurs many times within a small number of documents;
2. it is lower when the term occurs fewer times in a document, or occurs in many documents;
3. it is lowest when the term occurs in virtually all documents.

These measures are of interest for a first textual analysis of the competencies in the curricula.

8.2.2 Used Libraries

The automated processing of the curricula elements are done with the programming language *Python 3* which offers a high number of libraries for NLP and data science. For the text analysis, the Python library *NLTK* is used as it offers more flexibility and gives a better insight in the processes. Here, all steps of the preprocessing have to be selected and implemented individually. Listing 8.1 shows the source code for a typical preprocessing step in *NLTK* including tokenization, lemmatizing, and tagging.

For later processing, the library *spaCy* is used which combines several steps of preprocessing in one command. In listing 8.2 the basic commands for this library can be seen. At first a model, which is provided by *spaCy*, has to be loaded and connected to a command. In the example this command is called 'nlp' which can be applied to a string resulting in a preprocessed form of this string.

This command delivers an internal data structure with the following information for each token which are in this case terms [Spa20]:

Text: The original word text.

```

1 nlp = spacy.load("en_core_web_sm")
2 doc = nlp("The students are able to use information from the Internet
   in their work.")
3
4 for token in doc:
5     print(token.text, token.lemma_, token.pos_, token.tag_, token.dep_,
6           token.shape_, token.is_alpha, token.is_stop)

```

Listing 8.2: SpaCy library used for tokenization, tagging and lemmatizing

Lemma: The base form of the word.

POS: The simple part-of-speech tag.

Tag: The detailed part-of-speech tag.

Dep: Syntactic dependency, i.e. the relation between tokens.

Shape: The word shape – capitalization, punctuation, digits.

is alpha: Is the token an alpha character?

is stop: Is the token part of a stop list, i.e. the most common words of the language?

The results for the example in listing 8.2 are:

```

The the DET DT det Xxx True True
students student NOUN NNS nsubj xxxx True False
are be AUX VBP ROOT xxx True True
able able ADJ JJ acomp xxxx True False
to to PART TO aux xx True True
use use VERB VB xcomp xxx True False
information information NOUN NN dobj xxxx True False
from from ADP IN prep xxxx True True
the the DET DT det xxx True True
Internet internet NOUN NN pobj Xxxxx True False
in in ADP IN prep xx True True
their -PRON- PRON PRP$ poss xxxx True True
work work NOUN NN pobj xxxx True False
. . PUNCT . punct . False False

```

8.2.3 Necessary Adaptations

To compare and analyze the competencies of the seven selected educational models (see 3.5) using techniques from NLP, some adaptations are necessary as part of preprocessing. The first important adaptation considers the language aspects as four of the curricula are in English and three are in German language. To solve this problem two different methods are tested as preparation of the project:

- translate the German curricula into English with the support of the free translation software *DeepL*⁷ and manually review and preprocess afterwards, or
- preprocess the German curricula first based on German language models.

In preliminary test of the results of both solutions it turned out that the preprocessing models for English language deliver better results for the translated curricula than the models for German language do for the original curricula. Therefore, all German language curricula, so the Austrian competency model *digikomp4*, the German standards of the GI, and the curriculum of Switzerland, are translated into English language before preprocessing.

As already mentioned in section 4.6.2, five of the selected curricula use formulations starting like 'students are able to' but two, the Austrian competency model and the model from Berry for the English curriculum, use 'I can' statements. For text analysis the formulations using 'I can' prefixes are, like in the standardization process, changed into 'students are able to' form. That has no impact on analysis as the prefixes are handled as stop words and excluded during preprocessing.

8.2.4 Similarity Measures

In this approach the similarity of the formulations is used to automatically determine possible relations or even combinations between the competencies. Two popular methods for similarity measurement, the *Jaccard coefficient* and *cosine similarity* [HKP12] are applied and compared. The *Jaccard coefficient* is a measure for binary attributes. That means it compares the absence or presence of an element in an object which is denoted with 0 respectively 1. The *Jaccard coefficient* is calculated for two objects d_1 and d_2 as follows [HKP12, p. 71]:

⁷ <https://www.deepl.com/translator>

Definition 8.3 (Jaccard coefficient).

$$\text{sim}(d_1, d_2) = \frac{q}{q + r + s}$$

Where,

q is the number of attributes that equal 1 for both objects d_1 and d_2 ,

r is the number of attributes that equal 1 for object d_1 but equal 0 for object d_2 , and

s is the number of attributes that equal 0 for object d_1 but equal 1 for object d_2 .

In case of sentence similarities, the attributes are terms that occur in one or both compared sentences. If two sentences are completely similar the *Jaccard coefficient* has the value 1. The fewer terms the sentences share, the lower gets this value. This is a popular approach to determine text similarity but also suffers some issues as it does not consider the length of sentences and the repeated occurrence of one term in one sentence [HKP12].

Another popular similarity measure that considers the term frequency is the *cosine similarity* [HKP12]. It is based on *vector space models* which represent documents as 'vectors in a common vector space' [MRS08, p. 120]. One vector consists of weights for all terms in a dictionary. This weights can have the values of the *term frequency* [HKP12] or the *tf-idf* [MRS08]. So, for example, the comparison of the two competencies

CH-37.1: The students are able to write programs with loops [Leh14].

US2-10.3: The students are able to develop programs with simple loops, to express ideas [CST17].

would result in the vector displayed in table 8.1, using *term frequency* as weights, excluding stop words and showing lemmas in the dictionary.

The *cosine similarity* calculates the similarity of two documents d_1 and d_2 with their vector representations $\vec{V}(d_1)$ and $\vec{V}(d_2)$ as follows [MRS08, p. 121]:

Tab. 8.1: Vector for the two competencies CH-37.1 and US2-10.3 using term frequency

	write	program	loop	develop	simple	express	idea
CH-37.1	1	1	1	0	0	0	0
US2-10.3	0	1	1	1	1	1	1

Definition 8.4 (Cosine similarity).

$$\text{sim}(d_1, d_2) = \frac{\vec{V}(d_1) \cdot \vec{V}(d_2)}{|\vec{V}(d_1)| |\vec{V}(d_2)|}$$

Where,

the numerator represents the inner product of the vectors $\vec{V}(d_1)$ and $\vec{V}(d_2)$ which is defined for two vectors \vec{x} and \vec{y} as

$$\vec{x} \cdot \vec{y} = \sum_{i=1}^n x_i y_i$$

the denominator is the product of their Euclidean lengths which is for a vector $\vec{V}(d)$ of a document d , with n components $\vec{V}_1(d) \dots \vec{V}_n(d)$, defined to be

$$|\vec{V}(d)| = \sqrt{\sum_{i=1}^n \vec{V}_i^2(d)}$$

8.3 Results from Text Analysis in Curricula

8.3.1 Basic Text-Based Analysis

The basic text analysis for the curricula in English language, which are the Australian curriculum, the model for the English curriculum, and the CSTA standards from 2011 and 2017, done for this project, is described by Pasterk and Kesselbacher [PK18]. For their analysis the original versions of the curricula are used, before the standarization described in section 4.6.2 is performed. In case of the English curriculum the original teaching intentions [Nat14] are analyzed. Tab. 8.2 presents basic measures of the text analysis for all four curricula. It shows the mean, maximum, and minimum values for the *number of words* in one formulation and of the *word length* from the occurring words. The results prove that the teaching intentions of the English curriculum (GB) are very long and include a lot of words. This is an additional reason why the model from Berry [Ber15] is compared, analyzed and included in the *GGBMC* as it is mentioned in section 3.5.3. The formulations from the Australian curriculum (AU) show the second highest mean values and also the highest minimum value of 8. Both CSTA standards have similar values. Considering the word lengths, all four curricula show very similar values differing between 5.82 and 6.29 in the mean.

Pasterk and Kesselbacher describe the working process in the following words [PK18, p. 7]: After the extraction and the preprocessing of the learning outcomes

Tab. 8.2: Basic numbers of the formulations in the curricula (adapted from [PK18]).

	AC (AU)	EC (GB)	CSTA11 (US)	CSTA17 (US)
Number of words				
mean	17.19	23.44	13.73	14.79
max	25	48	28	25
min	8	5	5	6
Word length				
mean	6.14	5.82	6.29	6.03
max	16	15	17	15
min	1	1	1	1

we started with the main analysis. In a first step we wanted to collect information about the frequently used terms within the learning outcomes in each curriculum. The major goal was to use the gathered information to identify content and activity foci of the curricula. So we filtered the nouns, verbs, adjectives and adverbs from the learning outcomes and measured different values like the overall *term frequency* (tf_t), the number of learning outcomes containing a term (df_t), and the *tf-idf* ($tf - idf_{t,d}$) value which represents the importance of a word for a learning outcome within a curriculum. Listing 8.3 provides the source code to compute the *tf-idf* measures from a text file where each line represents one learning outcome. Note the preprocessing of the lines: white space and empty lines are removed, and punctuation characters are removed for better lemmatizing and tagging results. Also, to improve tagging, the omitted start phrase 'The students will be able to' is added as a prefix to each learning outcome before the start of the tagging process. The prefix phrase is removed after tagging so that it does not manipulate the results of the analysis. Then, the *TfidfVectorizer* performs fitting and transforming on the list of String learning outcomes. Ngrams of the size 1 (single words), 2 (bigrams) and 3 (trigrams) are considered. The *maximum document frequency* of considered words is half the corpus, and the *minimum document frequency* of considered words is 2 documents. The function from Listing 8.1 is used as *tokenizer*. Tokens are saved as *lemmas*, together with the tag category (similar tags are grouped, resulting in the tags *N* for nouns, *V* for verbs, *AJ* for adjectives, *AV* for adverbs and *O* for others). For the *tf-idf* computation, we exclude words with the tag for others, so that only relevant content words are considered. The following libraries were used in this

```

1 def tfidfPerLo(infile):
2     with open(infile) as f:
3         content = f.readlines()
4         content = [x.strip() for x in content]
5         content = filter(bool, content)
6         content = [x.translate({ord(i):None for i in string.punctuation
7                                }) for x in content]
8         tfidf = TfidfVectorizer(tokenizer=tokenizeTagLemmatize,
9                                stop_words='english', ngram_range=(1,3), max_df=0.5, min_df
                                =2)
10        tfs = tfidf.fit_transform(content)
11        return (tfidf, tfs);

```

Listing 8.3: Python function to compute the tf-idf measures for the learning outcomes of an unstructured text file using NLTK library [PK18].

approach: *NLTK WordNetLemmatizer*⁸, *Stanford POS Tagger*⁹ [TKMS03] and the *scikit-learn*¹⁰ python library for the computation of *tf-idf* measures.

Pasterk and Kesselbacher continue with the description of their proceeding [PK18, p. 8]: Key terms of each tag category can be found by computing the cumulative score of a term. Iterating over the learning outcomes, the *tf-idf* scores of one term are summed up for all learning outcomes. The terms with the highest cumulative scores are key terms in the respective category. Tab. 8.3 provides the results for computing the top 15% terms for each category of the Australian Curriculum. The most important nouns and verbs make it possible to determine the content focus (for nouns) and activity focus (for verbs) of a curriculum. With the resulting sets of key terms, the different curricula can be compared. Tab. 8.4, 8.5 and 8.6 display the five most frequent terms per tag. The frequency is determined by highest *tf* value, with higher *df* values acting as tiebreakers.

Pasterk and Kesselbacher describe the results for *nouns* as follows [PK18, p. 9]: Following these results, we assume that the curriculum in England (GB) focuses on programming as 'program' and also 'algorithm' are within the five most frequently used terms, but also 'content' plays an important role. For the Australian (AU) curriculum we assume that the focus is on data processing as 'data' and 'information' are within the five most frequently used terms, also the handling of algorithms and problems is an essential part. A similar situation can be found for the CSTA standards from 2017, where also 'data' and 'information' are within the five most frequently used terms. Additionally it includes 'program' and 'artifact' which shows that programming and the production or the handling of artifacts are of interest. In

⁸ <http://www.nltk.org/api/nltk.stem.html?highlight=lemmatizer>

⁹ <https://nlp.stanford.edu/software/tagger.shtml>

¹⁰ <https://scikit-learn.org/stable/>

Tab. 8.3: Overview of the top 15% terms (measured with cumulative *tf-idf* scores) of the analyzed Australian curriculum (adapted from [PK18]).

Group	Terms
Nouns (NN, NNP, NNPS, NNS)	data, system, digital system, information, algorithm, problem, user, range, solution, software, information system, program, requirement, account, idea, idea information, need, different type, different type data, type, type data
Verbs (VB, VBD, VBG, VBN, VBP, VBZ, MD)	explore, create, design, using, represent, investigate, are, use, recognise, implement, analyse, define, investigate how, including, design user, branching, involving
Adjectives (JJ, JJR, JJS, CD)	digital, digital system, simple, different, different type, different type data
Adverbs (RB, RBR, RBS, RP, WRB)	how, investigate how, how student, how student solution, explore how

Tab. 8.4: Comparison of the five most frequent nouns in each curriculum (adapted from [PK18]).

AC (AU)	EC (GB)	CSTA11 (US)	CSTA17 (US)
data	program	problem	program
system	technology	computer	data
information	content	software	problem
algorithm	(range)	technology	information
problem	system	information	artifact
	algorithm		

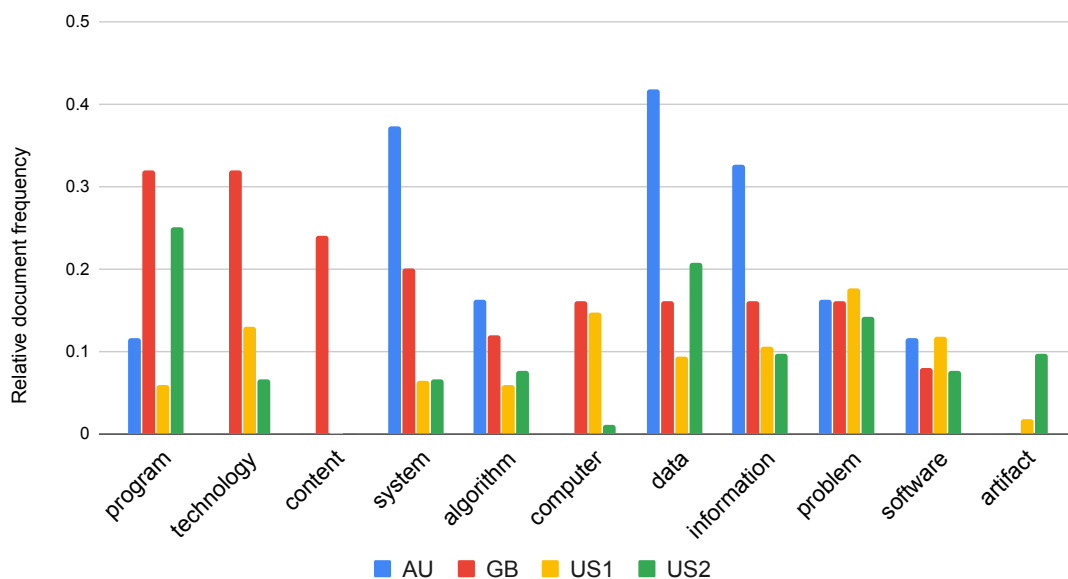


Fig. 8.1: Relative df of the five most frequently used nouns (adapted from [PK18]).

case of the CSTA standards from 2011 it can be assumed that the focus is on problem solution with the help of software, computers or other technologies. And again the term 'information' is important. The frequency of the terms is hard to compare because of the different size of the curricula and with that the different numbers of learning outcomes within the curricula. The relative comparison, calculated by the division of the df by the number of learning outcomes for each curriculum, is presented in Fig. 8.1. It is visible that the relative values are higher for the curricula with lesser learning outcomes (GB, AU). It has to be mentioned that Fig. 8.1 includes the most frequent terms of all curricula. Because of this, relative df values are shown for more terms compared to Tab. 8.4. Fig. 8.2 shows the impact of the collected five most frequently used terms in all curricula. The terms 'algorithm' and 'information' are very important to the curriculum of Great Britain (GB). For the CSTA standards from 2011 and 2017 (US1 and US2) the term 'artifact' is very important. For the CSTA standards from 2011 (US1) the terms 'algorithm' and 'program' and for the CSTA standards from 2017 (US1) 'information' and 'data' have the next highest values. In the Australian curriculum (AU) all used terms have a similar mean $tf-idf$ value.

Verbs are also part of the discussion from Pasterk and Kesselbacher [PK18, p. 11]: The five most frequently used verbs, important to collect information about the activity focus of curricula, are presented in Tab. 8.5. In all curricula the number of appearances of the term 'use' is the highest of all verbs (following tf values first

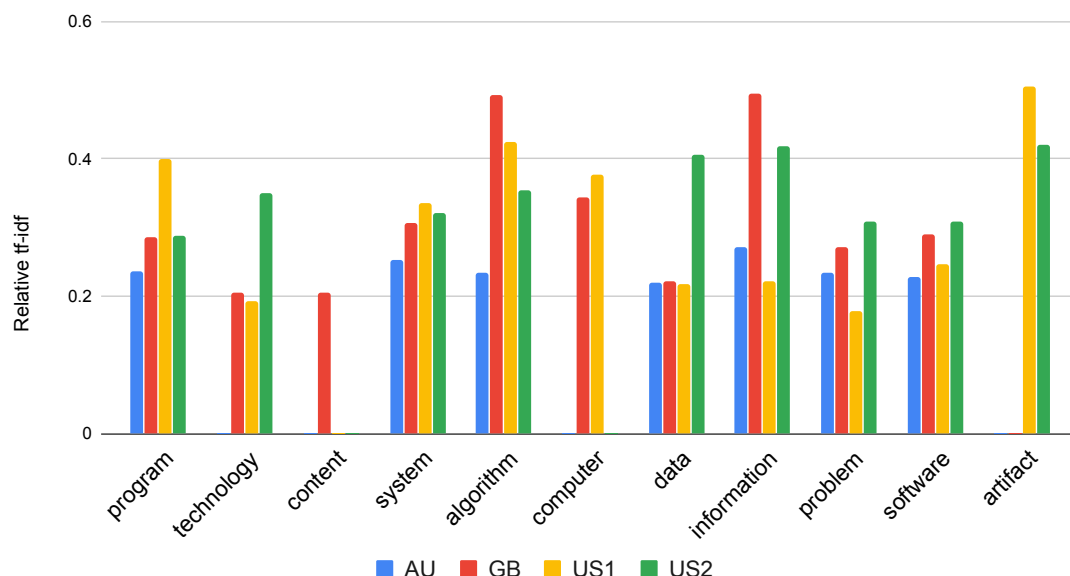


Fig. 8.2: Mean $tf-idf$ value of the five most frequently used nouns for each curriculum (adapted from [PK18]).

and then the df values to break ties). So it can be assumed that curricula for information technology related subjects focus on the usage of undefined objects or elements like devices, software, concepts, or others. Further the term 'create' can be found in three of the four curricula which indicates that the creation of products is very important. Descriptive activities are also important as the term 'describe' can be found in two curricula. Differences can be found as well. The verb 'understand' is only frequently used in GB, 'explore' and 'represent' appear only in AU, and 'solve' and 'demonstrate' can only be found in US1. Also, in Fig. 8.3, it is visible that the term 'use' has very high relative df values in all curricula, compared to the other verbs. Only for the model for the curriculum from England (GB) the term 'understand' has a comparable relative df value. Additionally Fig. 8.3 shows that in smaller curricula, like the one from England (GB), the frequently used terms appear more often than in larger curricula. As it is visible in Fig. 8.4, the term 'use' has a very low mean $tf-idf$ value compared to the other most frequently used terms. That is because the term is used so frequently that it loses importance for single learning outcomes. The highest mean $tf-idf$ values can be found for the terms 'explore' and 'represent' in the CSTA standards from 2011. Compared to Fig. 8.3, these terms have very low relative df values which makes them more important for the single learning outcomes. The most important verbs for the English curriculum are 'create' and 'develop', for the Australian curriculum they are 'solve' and 'design', and for

Tab. 8.5: Comparison of the five most frequent verbs in each curriculum (adapted from [PK18]).

AC (AU)	EC (GB)	CSTA11 (US)	CSTA17 (US)
use	use	use	use
create	understand	describe	create
represent	create	identify	develop
design	design	solve	describe
explore	develop	demonstrate	explain

the CSTA standards from 2017 they are 'represent' and 'explain'.

Finally, Pasterk and Kesselbacher analyze *adjectives* and *adverbs* as follows [PK18, p. 12]: Besides the nouns and verbs we also investigated whether the adjectives and adverbs can add information about the focus of a curriculum. Therefore we also considered the five most frequently used terms of these two categories. The frequency is determined by highest *tf* values first, and then the *df* values to break ties. The results can be found in Tab. 8.6. The term 'digital' is in three curricula under the five most frequently used adjectives. That is not surprising, considering the overall topic of the curricula. Also the adjective 'simple' can often be found in three different curricula. That indicates content or concepts which had to be simplified so that the learning outcomes are not that challenging. The adjective 'appropriate' appears in two curricula which can be interpreted as freedom for the teacher to decide which is 'appropriate'. More interesting are those adjectives that distinguish the curricula from each other. Knowing that the GB uses terms like 'binary', 'computational' and 'logical' one could assume that this curriculum focuses on mathematical operations. With the background knowledge of this curriculum this assumption can not be verified. From the most frequent adjectives of the AU curriculum it can only be assumed that a social aspect is important. In the CSTA17 standards the most frequent adjective is 'computational' but without information about the context it is hard to make assumptions. The CSTA11 standards include three adjectives which do not appear that frequently in the other curricula: 'collaborative', 'ethical', and 'mobile'. After the adjective analysis we can draw the conclusion that single adjectives only add little information about the focus of the curricula. Adjective-noun combinations would give additional insights. The adverbs on their own also seem to add only little information. It can be assumed that the collaboration plays an important role in at least three of the curricula. For the GB curriculum it is important to handle things respectfully, safely, and responsi-

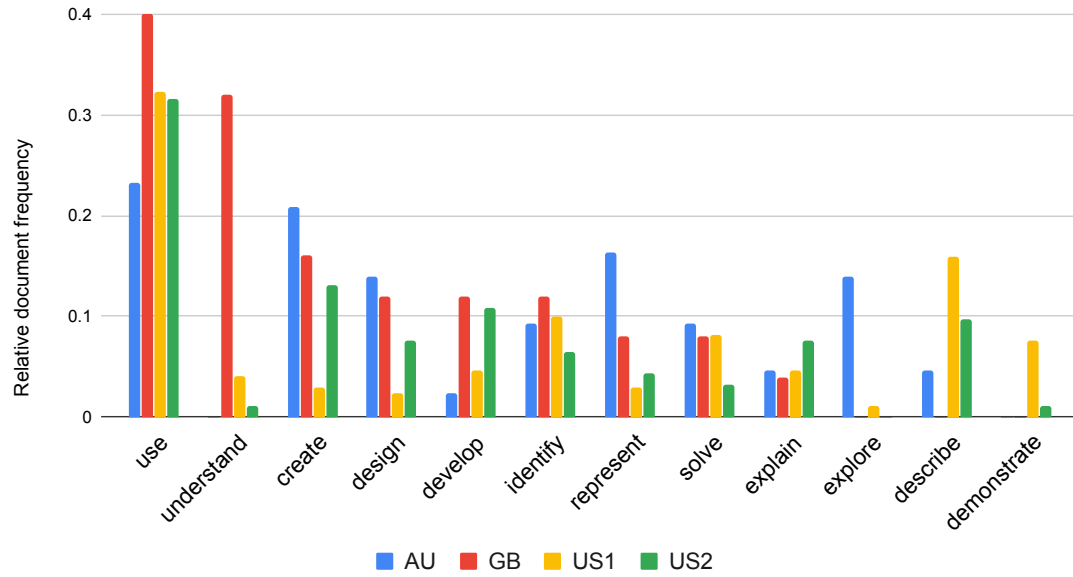


Fig. 8.3: Relative df of the five most frequently used verbs (adapted from [PK18]).

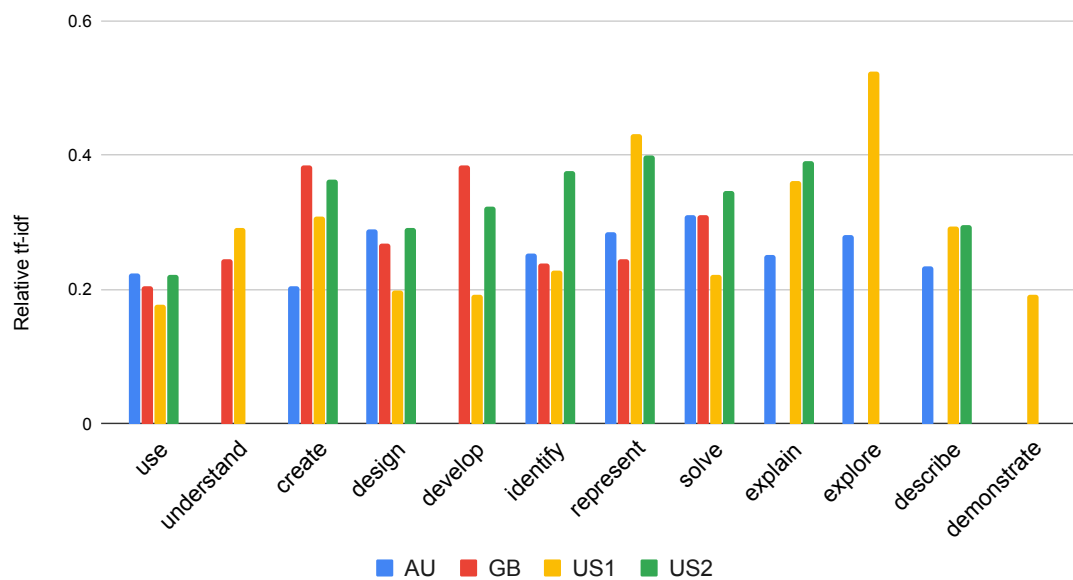


Fig. 8.4: Mean $tf-idf$ value of the five most frequently used verbs for each curriculum (adapted from [PK18]).

Tab. 8.6: Comparison of the five most frequent adjectives and adverbs in each curriculum (adapted from [PK18]).

AC (AU)	EC (GB)	CSTA11 (US)	CSTA17 (US)
adjectives			
digital	digital	appropriate	computational
simple	binary	digital	different
different	simple	collaborative	appropriate
social	computational	mobile	multiple
common	logical	ethical	personal
adverbs			
collaboratively	respectfully	developmentally	systematically
diagrammatically	safely	collaboratively	iteratively
independently	responsibly	accurately	appropriately
creatively	digitally	computationally	clearly
critically	effectively	cooperatively	collaboratively

bly. Besides collaboration the AU also addresses independent and individual work. The CSTA17 standards focus on systematic and clear activities, where the CSTA11 prefer development and accuracy.

8.3.2 Terms Occurrences in Categories

In the previous section the terms occurring in the curricula are analyzed in terms of the numbers of their occurrences. To support the automation of the categorization the occurrences of terms in the categories for computer science education [DB15] is of great interest. As the *nouns* are the part of speech that is decisive for the categorization, in this section the results for the analysis of the *noun* occurrences are discussed. Overall 371 different *nouns* are identified in the seven selected curricula. Taking a detailed look at the occurrences it shows that a great part of the *nouns* (65.8%) appears only in one category. In two categories occur 21.3%, in three 8.9%, in four 3%, and only 1.1% of the terms occur in all five categories [Chy20]. Fig. 8.5 shows this distribution of occurrences.

Four nouns occur in all five categories. These are 'data', 'information', 'range', and 'system'. Those eleven nouns appearing in four categories are in nine cases not

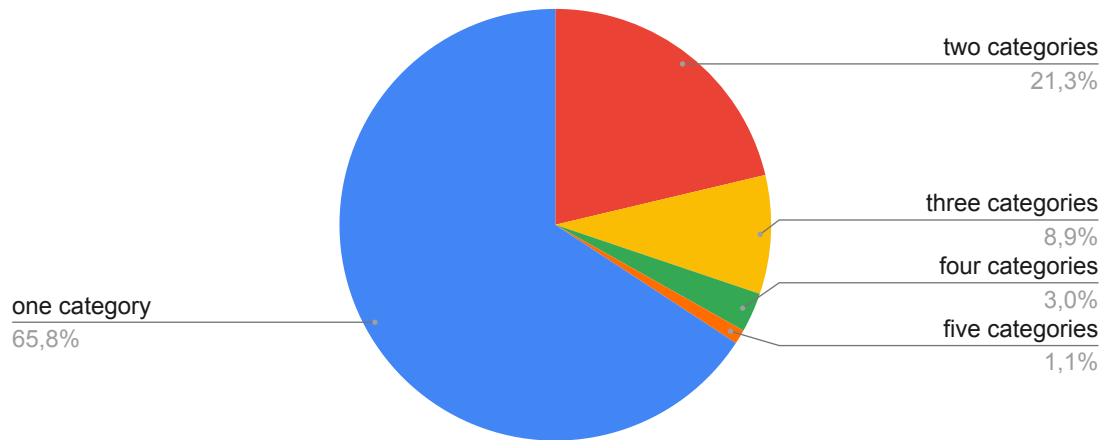


Fig. 8.5: Occurrence of nouns in the categories (adapted from [Chy20]).

found in the category *data representation* which includes the lowest number of competencies (see section 5.5). 33 *nouns* occur in three categories. The combinations of categories which share the most *nouns* are *human factors and ethics*, *digital infrastructure*, and *programming and algorithms* with 9 shared *nouns*, *human factors and ethics*, *digital applications*, and *programming and algorithms* with 8 shared *nouns*, and *human factors and ethics*, *digital applications*, and *digital infrastructure* with 6 shared *nouns*. With 26 *nouns* from the category *human factors and ethics*, 21 from *digital applications*, 21 from *digital infrastructure*, and 23 from *programming and algorithms* the *nouns* which occur in three categories are equally distributed over these four categories. Only the category *data representation* shares with 8 only few *nouns* with two other categories. 79 of the *nouns* are shared by two categories. With 16 common *nouns* the categories *human factors and ethics* and *digital applications* share the most, followed by *human factors and ethics* and *digital infrastructure* with 15, and *digital applications* and *programming and algorithms* with 14. Here, again the four categories *human factors and ethics*, *digital applications*, *digital infrastructure*, and *programming and algorithms* are represented with similar numbers which are 40, 39, 30, and 36 *nouns*. The category *data representation* shares 13 *nouns* with one other category. As no direct assumptions of correct category are possible with 2, 3, 4, and 5 shared *nouns*. Together these are 34.3% of nouns which can not be used for the categorization. But the results can be used to give experts suggestions for suitable categories [Chy20].

The unique *nouns* are with the number of 244 (65.8%) the by far largest group. It can be assumed that this group has the highest chances to support a correct categorization. Fig. 8.6 presents their distribution over the five categories. In this case 28.3% (69 *nouns*) of the unique *nouns* are found in the category *human factors*

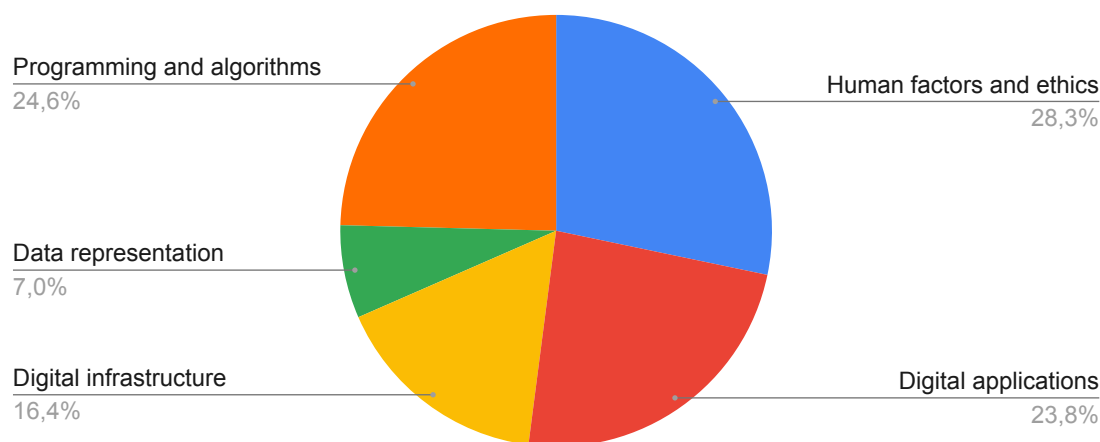


Fig. 8.6: Occurrence of unique nouns in the categories (adapted from [Chy20]).

and ethics, 24.6% (60 nouns) in programming and algorithms, 23.8% (58 nouns) in digital applications, 16.4% (40 nouns) in digital infrastructure, and 7% (17 nouns) in data representation [Chy20].

This detailed analysis of terms does not only show the frequency and occurrences of the terms but also that there are different words for the same meaning. This happens quite often and makes an automated approach difficult. To improve this situation as much as possible, *dictionaries*, including *synonyms* for the occurring terms, are created. The process of the generation of these *dictionaries* and their use are described in the following section.

8.4 Synonym Replacement

8.4.1 Building a Dictionary

To improve the similarity measures that play a central role in this approach, dictionaries are built to support the identification of synonyms and to store different forms of the words. The process is described by Verkhovets and is, in a first attempt, done manually because existing synonym lists are often missing some important items in the context of computer science curricula [Ver19]. First steps of this process are characterized by reading and analyzing the seven selected curricula and extracting found key *nouns* and *verbs*. Words which are not relevant for the curricula and often appear in the examples included in the competency formulations are not considered for the dictionaries. This phase results in a list of alphabetically sorted *nouns* and a list of alphabetically sorted *verbs*. Based on these two lists the terms are analyzed

```

1 <term>
2   <lemma>algorithm</lemma>
3   <singular>algorithm</singular>
4   <plural>algorithms</plural>
5   <synset>algorithm; algorithms | sequence of instruction; sequences
      of instructions | sequence of step; sequences of steps | set of
      step-by-step instruction; sets of step-by-step instructions</
      synset>
6   <def>a step-by-step procedure for solving a problem</def>
7 </term>

```

Listing 8.4: Example of a dictionary entry for *nouns* [Ver19].

and *synonyms* are identified. If possible, online lists like *Woxikon*¹¹ or *Wordnet*¹² are used to support this process. In some cases only two related words are found, but in general there are more synonyms. For *verbs* larger groups of related terms are found than for *nouns*. The largest group contains six related words which are 'create', 'design', 'establish', 'generate', 'initiate', and 'invent'. The synonym terms are collected to sets of synonyms, so called *synsets*, sorted in alphabetical order for easier processing. All terms are stored following the structure of *Extensible Markup Language (XML)* [Ver19]. During this process 36 *nouns* and 88 *verbs* are collected in the dictionary. An example for a *noun* can be found in Listing 8.4, whereas Listing 8.5 shows an example for *verbs*. In both dictionaries the entries are tagged as `< term >` and contain the tags `< lemma >`, `< synset >`, and `< def >`. Under the `< lemma >` tag the lemma of the term is stored, `< synset >` includes a set of synonyms in different forms, and `< def >` gives a definition for the term retrieved from online dictionaries like the *Cambridge Online Dictionary*¹³. *Nouns* also include the tag for singular (`< singular >`) and plural (`< plural >`) forms. For *verbs* tags for infinitive (`< infinitive >`), gerund (`< gerund >`), and past participle (`< past – participle >`) forms are provided.

After the structured collection of synonyms, one *core synonym* for each *synset* is selected which represents the *synset* in the similarity measurement. This selection depends on the 'frequency of usage' and 'breadth and neutrality of meaning' of the terms [Ver19]. The definitions of the terms from online dictionaries like the *Cambridge Online Dictionary* support the selection.

¹¹ <https://synonyms.woxikon.com/>

¹² <https://wordnet.princeton.edu/>

¹³ <https://dictionary.cambridge.org/>

```

1 <term>
2   <lemma>create</lemma>
3   <infinitive>create</infinitive>
4   <gerund>creating</gerund>
5   <past-participle>created</past-participle>
6   <synset>create; creating; created | construct; constructing;
      constructed | develop; developing; developed | generate;
      generating; generated | invent; inventing; invented | produce;
      producing; produced | write; writing; written</synset>
7   <def>to make something new, or invent something</def>
8 </term>

```

Listing 8.5: Example of a dictionary entry for *verbs* [Ver19].

8.4.2 Using Dictionaries in Implementation

The described dictionaries are used in the automation of the relations generation and the combination to *intersector nodes* as they support the similarity measurement. In the implementation they play a major role as they help standardizing the formulations on the basis of used words. The idea is to substitute all terms which appear in the dictionaries by the *core synonym* from their *synset*, as Chystopolova describes [Chy19]. So, in a first step, the information from the dictionaries is imported in form of a multidimensional list. For a better performance during the search in the dictionaries an index is generated, based on the first letter of the words and the positions of corresponding entries in the multidimensional list. The competency lists are also imported and preprocessed using the *spaCy* library. This library automatically provides the *lemmas* for all terms in the competency formulations. The *lemmas* are compared to the entries in the corresponding index, based on their first letter. If a *lemma* is found in the dictionary, it is checked if it is a *core synonym*. In the case of a positive check the term stays the same. Otherwise it is replaced by the *core synonym* in the corresponding *synset*. This whole process delivers CSV files which contain the original competencies, the standardized competencies, and lists of *nouns*, *verbs*, *adjectives*, and *adverbs* [Chy19].

8.5 Generation of Relations

8.5.1 Similarity Measure Comparison

The identification of the relations between the competencies is one of the aims of this natural language processing based approach. But, it is a very difficult task if all possible relations are considered as in a graph G with n nodes there exist $n * (n - 1)$ possible edges. These are in the case of the smallest standardized curriculum

(the model for the English curriculum) with 62 competencies already 3,782 possible dependencies.

The idea behind this approach is that the dependencies exist between competencies which are linguistically similar. In case of the *expands* relation this assumption seems to be valid as they cover the same topics. But *requires* relations can be from total different areas and do not include similar words. Nonetheless, both relations are considered in the following experiments as they are both important for the graph-based approach (see chapter 4). After the evaluation considering both relations, an additional analysis distinguishing the two relations is presented. The *Jaccard coefficient* and *cosine similarity* based on *tf-idf* values are used to determine the similarities between the competencies. The results are compared to the relations evaluated by experts (see section 4.5.3) and with 'best' results those results are meant which are closest to the experts opinions.

In a first experiment both methods are compared to each other to determine the more suitable method for this kind of data. Both methods require a threshold which determines up to which similarity values the relations should be considered. Some prior tests show that for the *Jaccard coefficient* the limit with the best results is 0.88, as total completeness is 1, and for *cosine similarity* it is 0.3. As input data the standardized competency formulations (as described in section 4.6.2) without synonym replacement (see section 8.4) are used. The following part of speech combinations are tested and also compared:

- all words
- all words without stop words
- nouns, verbs and adjectives
- nouns and verbs
- nouns and adjectives
- nouns

Both methods are compared with all combinations in terms of *precision* and *recall* which are frequent measures for the effectiveness of information retrieval methods and are defined as follows [MRS08, p. 155]:

Definition 8.5 (Precision). ***Precision** (P) is the fraction of retrieved documents that are relevant*

$$Precision = \frac{\#(relevant\ items\ retrieved)}{\#(retrieved\ items)} = P(relevant|retrieved)$$

Tab. 8.7: A contingency table to calculate *precision* and *recall* [MRS08, p. 155].

	Relevant	Nonrelevant
Retrieved	true positives (tp)	false positives (fp)
Not retrieved	false negatives (fn)	true negatives (tn)

Definition 8.6 (Recall). *Recall (R)* is the fraction of relevant documents that are retrieved

$$Recall = \frac{\#(\text{relevant items retrieved})}{\#(\text{relevant items})} = R(\text{retrieved}|\text{relevant})$$

Manning, Raghavan, and Schütze suggest a contingency table to be used to calculate both measures which is structured as shown in Tab. 8.7 [MRS08, p. 155]. With the help of Tab. 8.7 *precision (P)* and *recall (R)* can be calculated as follows [MRS08, p. 155]:

$$P = tp/(tp + fp)$$

$$R = tp/(tp + fn)$$

To evaluate the results from the similarity measures the contingency table is filled with found dependencies which are also considered by experts (as *tp*), not found dependencies which are considered by experts (as *fn*), found noise dependencies which are not considered by experts (as *fp*), and not found dependencies which are also not considered by experts (as *tn*). This results in tables for every curriculum representing each method and part of speech combination. Because of the high number of generated tables, not all of them can be discussed here. To show a sample of the results the values for the CSTA standards from 2011 are presented and compared. The most promising part of speech combination for both methods is in this case *nouns and verbs*. All other combinations lead to lower numbers of found dependencies which are also considered by experts (as *tp*). Tab. 8.8 shows the contingency table including the results from *Jaccard coefficient* and *cosine similarity* for the CSTA standards from 2011. 28 from the 122 relations considered by experts are found. Additionally 62 relations, which are not considered by experts, are identified. As it is visible in Tab. 8.8 the results from the *cosine similarity* based on the *tf-idf* are very similar to the values from *Jaccard coefficient*. 24 of the 122 relations considered by experts are found. 41 relations are identified which are not considered by experts.

As it can be seen the results are very close which is also reflected by the values for *precision* and *recall*.

Jaccard coefficient: $P = 0.31$, $R = 0.23$

Tab. 8.8: The contingency table for the CSTA11 standards using *Jaccard coefficient* and *cosine similarity* with *nouns and verbs*.

Jaccard coefficient		
	Considered by experts	Not considered by experts
Found dependencies	28 (tp)	62 (fp)
Not found dependencies	94 (fn)	10,322 (tn)
Cosine similarity		
	Considered by experts	Not considered by experts
Found dependencies	24 (tp)	41 (fp)
Not found dependencies	98 (fn)	10,343 (tn)

Cosine similarity: $P = 0.37$, $R = 0.20$

Here, the *Jaccard coefficient* has a slightly higher *recall* (R), whereas the *precision* (P) is higher with *cosine similarity*. If the decision which of the two similarity measures should be used in further calculations would only depend on this curriculum, the *cosine similarity* would be preferred as the value for *precision* is higher. But, taking a look at the results from all other curricula it shows that overall the *Jaccard coefficient* provides results which are closer to the experts' opinions. These results are calculated by summing up all values for *tp*, *fp*, and *fn* and determine *precision* (P) and *recall* (R) for these sums which leads to the following values:

Jaccard coefficient: $P = 0.29$, $R = 0.25$

Cosine similarity: $P = 0.27$, $R = 0.22$

In this case both values are higher applying the method using the *Jaccard coefficient*. Therefore, this measurement is selected together with the part of speech combination *nouns and verbs* for further experiments.

8.5.2 Improvement of Results

As it is planned that this tool proposes to the experts relations of competencies, the question arises which aspect of the evaluation should be improved. This can be answered by thinking about the process of evaluation. It is easier for an expert to

rate a given relation between to competencies as *not okay* than to add a relation that is missing. That means the goal of improvement is to increase the *recall* (R) value as this represents the ratio of found and not found relations which are considered by experts. On the other hand the *precision* (P) should not decrease too much as this would mean that the number of found relations, which are not considered by experts, increases. With regard to these criteria, the improvement is intended.

For a second attempt only the similarity measurement with the more promising results from the first experiments, the *Jaccard coefficient*, is used. In contrast to the first attempt the process itself and the preprocessing are adapted to improve the results. As an additional preprocessing step the synonym replacement (as described in section 8.4) is applied. With this, more similar words are found which leads to an improvement of similarity measures [Chy19]. The similarity measurement using the *Jaccard coefficient* is modified by the addition of *weights*. These *weights* affect the results in a way that more important terms have a higher *weight* and therefore increase the similarity. In the calculation of the *Jaccard coefficient* all occurring terms are denoted with a value 1, where not occurring terms have a value 0. *Weights* can be added by changing the value of occurring words. So, instead of giving occurring terms the value 1, they get e.g. the value 0.8. A *weight* smaller than 1 shows a lower importance of the terms. Several tests show that in case of the curriculum data the weighting of terms leads to closer results to the experts opinions. For this approach a weighting system is developed that depends on the presence of different part of speech (POS) words. The highest weight is given to *nouns*, followed by *verbs*, and *adverbs*, *adjectives*, and *auxiliary words* at the same level. *Nouns* and *verbs* occur in every formulation, whereas it can happen that there are no *adverbs*, *adjectives*, or *auxiliary words*. All cases have different distributions of the weights [Chy19] which can be seen in Tab. 8.9.

The described steps led to new results for each curriculum. To enable a comparison to the results from the first experiment, the results for the CSTA standards from 2011 can be found in Tab. 8.10. It includes the results for the three different thresholds for the *Jaccard coefficient* 0.5, 0.398, and 0.23. The results on this level are very similar. Decreasing the threshold means an increase of *tp* but also of *fp*. Comparing the results with a threshold of 0.5 and 0.398 the difference is 1 in *tp* and 2 in *fp*. But from 0.398 to 0.23 there are 2 more cases of *tp* and 55 more cases of *fp*. This is a very high increase for two more found relations which are considered by experts.

The *precision* (P) and *recall* (R) values are also very close in this curriculum.

Jaccard coefficient 0.5: $P = 0.30$, $R = 0.41$

Jaccard coefficient 0.398: $P = 0.30$, $R = 0.42$

Jaccard coefficient 0.23: $P = 0.23$, $R = 0.43$

Tab. 8.9: Weights for the different cases of occurring POS elements [Chy19].

Part of speech	Weights			
	All	Without AW	Without Adj and AW	Without Adv, Adj, and AW
Nouns	0.50	0.53	0.53	0.58
Verbs	0.35	0.37	0.37	0.42
Adverbs	0.05	0.05	0.10	0
Adjectives	0.05	0.05	0	0
Auxiliary words	0.05	0	0	0

Tab. 8.10: The contingency table for the CSTA11 standards using modified *Jaccard coefficient* with *nouns and verbs*.

Jaccard coefficient: 0.5		
	Considered by experts	Not considered by experts
Found dependencies	50 (tp)	116 (fp)
Not found dependencies	72 (fn)	10,268 (tn)
Jaccard coefficient: 0.398		
	Considered by experts	Not considered by experts
Found dependencies	51 (tp)	118 (fp)
Not found dependencies	71 (fn)	10,266 (tn)
Jaccard coefficient: 0.23		
	Considered by experts	Not considered by experts
Found dependencies	53 (tp)	173 (fp)
Not found dependencies	69 (fn)	10,211 (tn)

Changing the threshold from 0.398 to 0.23 means an increase for *recall* (R) of 0.01 but with that a decrease of the *precision* (P) of 0.07. This difference seems to be not that high but considering 55 additional relations the threshold of 0.398 is the most valuable choice. Therefore, the results for this threshold are compared to those of the *Jaccard coefficient* from the first experiment:

Without synonym replacement and weights: $P = 0.31$, $R = 0.23$

With synonym replacement and weights: $P = 0.30$, $R = 0.42$

As the threshold in the second experiment is lower, the value for *precision* (P) is also lower (0.01). But this difference is negligible as the value for *recall* (R) increases in the second experiment by 0.19. This is a great improvement considering the small decrease of *precision*.

Taking a look at Tab. 8.11 presenting the overall values, which are again the sums of the values for *tp*, *fn*, and *fp*, the same trend as in the CSTA standards from 2011 is visible. The decrease of the threshold from 0.5 to 0.398 has the consequence of 33 more *true positives* (*tp*) and 152 more *false positives* (*fp*). Comparing the results from 0.398 and 0.23 the difference of *tp* is 67 and of *fp* it is 822. This is a very high additional number of by experts not considered relations. The difference can also be recognized in the *precision* (P) and *recall* (R) values:

Jaccard coefficient 0.5: $P = 0.28$, $R = 0.40$

Jaccard coefficient 0.398: $P = 0.26$, $R = 0.44$

Jaccard coefficient 0.23: $P = 0.18$, $R = 0.54$

Especially the value of *precision* (P) drops very low in case of a threshold of 0.23. Considering the very large number of 822 additional relations which are not correct following the experts' opinions, again the *Jaccard coefficient* with a threshold of 0.398 is the best option. These values can be compared to those of the first experiment:

Without synonym replacement and weights: $P = 0.29$, $R = 0.25$

With synonym replacement and weights: $P = 0.26$, $R = 0.44$

Again, a negligible decrease of 0.03 can be recognized for *precision* (P). But in case of *recall* (R) again an increase of 0.19 is visible in the second experiment. This means an overall improvement by adding synonym replacement and weights into the comparison process.

A detailed analysis of the results for the identification of the dependency relations confirms the assumption that the similarity measures only detect *expands* relations.

Tab. 8.11: The contingency table for the overall results from modified *Jaccard coefficient* with *nouns and verbs*.

Jaccard coefficient: 0.5		
	Considered by experts	Not considered by experts
Found dependencies	287 (tp)	756 (fp)
Not found dependencies	435 (fn)	69,138 (tn)
Jaccard coefficient: 0.398		
	Considered by experts	Not considered by experts
Found dependencies	320 (tp)	908 (fp)
Not found dependencies	402 (fn)	68,986 (tn)
Jaccard coefficient: 0.23		
	Considered by experts	Not considered by experts
Found dependencies	387 (tp)	1730 (fp)
Not found dependencies	335 (fn)	68,164 (tn)

In the list of the found relations for all curricula no single *requires* relation considered by the experts is included [Chy19]. If it is assumed that only the *expands* relations can be found by similarity measures and only these relations are considered in the calculation of *precision* and *recall*, the overall value for *recall* increases from 0.44 to 0.53. Taking a look at the values from the curricula, Tab. 8.12 shows that all three models translated from German language have a lower *recall* than the English ones. All English language curricula have a *recall* over 0.55, whereas the values of all German language curricula are below 0.50. That means it has an impact on the relation determination if English or non English language curricula are included.

An additional analysis of the data shows that approximately 30% of the *expands* relations considered by experts connect two competencies which contain no common words. So, by further improving this approach, in the best case a maximum of 70% of the *expands* relations considered by experts can be found [Chy19]. For the identification of *requires* relations it needs additional techniques which will be added

Tab. 8.12: *Precision* and *recall* considering only *expands* relations.

Number of	DK (AT)	AC (AU)	21 (CH)	GI (DE)	EC (GB)	CSTA (US)	
						2011	2017
Precision	0.35	0.18	0.32	0.24	0.28	0.30	0.29
Recall	0.47	0.56	0.49	0.47	0.59	0.58	0.57

in future projects [Chy20].

Two aspects of the automated generation of relations also have to be mentioned. At first, the found relations are not directed. The similarity measures only show if there is a relation between two competencies or not. In a first approach the directions are determined with the help of *Bloom's revised taxonomy* and defined action verbs (see section 2.5). It follows the idea that the verbs of two similar competencies belong to two different cognitive levels as defined by Anderson et al. [AK01]. In such a case the direction shows from the competency with the higher level verb to the competency with the lower level verb. But, it turns out that most competency formulations use verbs on the same cognitive level [Chy20] which results in very few found directions. A consideration for future work will be transitive relations. It shows that a lot of the found dependencies which are not considered by experts are transitively related. That is why they are very similar. In case of the CSTA standards from 2011 (from the 118 *false positives* (*fp*) with an threshold of 0.398 (see Tab. 8.10)) only eight are not transitively related [Chy20]. In the analysis described in this section this fact is not considered. That means the majority of *false positives* are indirectly related and therefore not considered by experts.

8.6 Automated Categorization

8.6.1 Introduction

For the generation of the *Generic, Graph-Based Model for Competencies (GGBMC)* (see section 4.6) categorization is an important part. In this step competencies are grouped which belong to the same topic. The results give first hints which competencies are to be combined. To automate this process two independent experiments are conducted based on different approaches. The first approach is connected to the categorization into *computer science* (*CS*) and *digital literacy* (*DL*) which is discussed in section 5.5. The second one is based on the linguistic analysis of the curricula described in section 8.3.2. In a survey described in section 5.5 nine experts categorized competencies into *CS*, *DL*, *both*, or *none*. The detailed results of

this survey are also discussed in section 5.5. As preparation for the automation of this process the nine experts are asked for their strategies. Pasterk, Kesselbacher and Bollin discussed the results in the following words [PKB19, p. 5]: Considering the answers of the experts regarding their strategy, seven out of the nine experts refer to the definitions of CS or DL. Six experts use keywords that they assign to either *CS* or *DL*. Finding keywords or key terms was the way to categorize for two experts. Two other experts focused on the topics of the learning outcomes and the combined objectives which are often defined by keywords. Eight out of nine experts take keywords in account during categorization.

These results show that keywords occurring in the competency formulations are, besides the definitions of the categories, the most crucial elements for a categorization process. In the following two sections two different and independent approaches to automatically retrieve lists of representative keywords, which are in the following called *dictionaries*, are discussed.

8.6.2 Computer Science or Digital Literacy

The process and results of the automated categorization into *CS* and *DL* is already discussed by Pasterk, Kesselbacher and Bollin [PKB19]. For the categorization into *CS* and *DL* four of the seven curricula are selected to be analyzed. These are the Australian curriculum, the Curriculum from Switzerland, and the CSTA standards from 2011 and 2017 in their original versions.

Pasterk, Kesselbacher and Bollin start with the description of the process as follows [PKB19, p. 6]: We now present an automated categorization approach based on linguistic features to assign a category, either *CS* or *DL*, to each learning outcome of the four analyzed curricula. The categorization results are evaluated against the expert classification (see section 5.5). The analyzed curricula are available as PDF documents. Manual preprocessing was done by extracting the texts of the learning outcomes. The extraction is implemented in Python. The process of extraction of the linguistic features includes the following basic techniques: normalization of words to improve comparability (lowercase, lemmatizing), stop word removal with a list of English stop words, word tokenizing to produce term lists. Each learning outcome text constitutes a single element, called document, in the analysis. Tagging is applied with a trained part-of-speech tagger [TKMS03] to extend the words with part of speech categories. The learning outcome text is tagged in full sentence form - 'the students will be able to' is added at the beginning. Tags are grouped; one of the following categories is assigned to each word: noun, verb, adjective, adverb, and other. After tagging, the sentence start is removed, and the tag category and lemmatized words are stored as term lists for each learning outcome. For the categorization, linguistic frequency measures of curricula representative for *CS* and *DL* education are computed, following the same linguistic processing, and stored

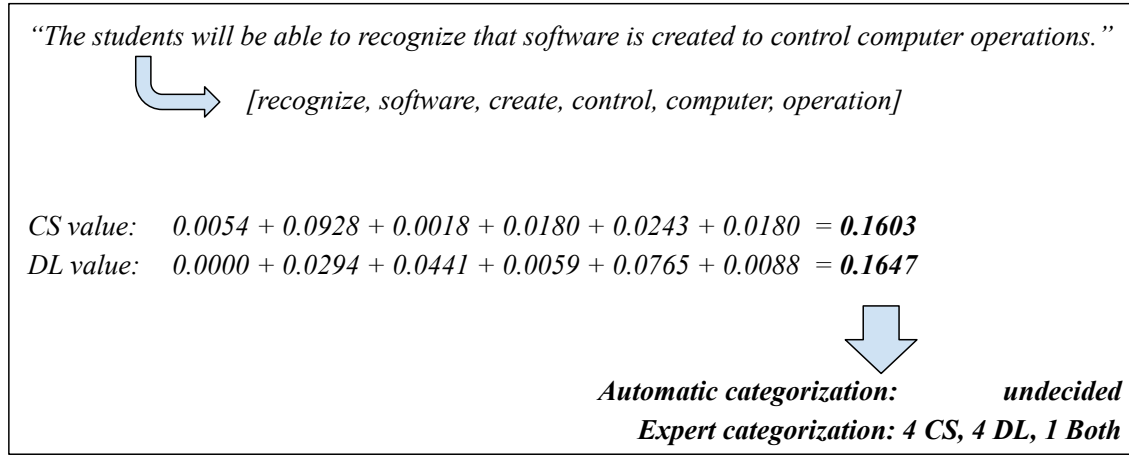


Fig. 8.7: Example for learning outcome processing from CSTA standards from 2011 [PKB19].

in two dictionaries. The *Computer Science Curricula 2013* (AIE) [JTFoCCS13b], created by a cooperation of ACM and IEEE members, contains a set of curriculum guidelines for undergraduate *CS* programs and is used to build the dictionary for *CS*. For *DL*, the dictionary is built from three curricula. *The Digital Competence Framework for Citizens (Dig)* [CVP17] designed by the European Union in 2017, *British Columbia Digital Literacy Framework (BC)* [oBC13] from the Province of British Columbia, and *digikomp4 (DK)* [Dig13b] from the Austrian initiative for digital competencies and informatics education. The considered linguistic features include *term frequency (tf)*, *term frequency over inverse document frequency (tf-idf)* and *document frequency (df)* [HKP12]. The metrics *tf* and *tf-idf* performed poor for the categorization because of size differences in the dictionaries. For categorization, the *df* value is used. In this context, this value describes in which fraction of learning outcomes a term occurs. For each learning outcome to be categorized, the sum of the *df* values of the occurring tagged terms in the two dictionaries is computed. Insights from the experts’ strategies suggest that content terms (*nouns*), cognitive activity terms (*verbs*) and their combinations should be considered. In this contribution, individual terms tagged as *nouns* and *verbs* are counted. The highest value determines the category. When both sums are within 10% of the highest value, a third category ‘*undecided*’ is assigned. Fig. 8.7 shows an example of this process.

Pasterk, Kesselbacher and Bollin continue with a comparison of the categorization results to experts classification [PKB19, p. 6]: The automated categorization results are compared to the expert classifications in two ways. First, the categorization is compared against the expert classification regarding all learning outcomes of the curricula. The results show the categorization performance for a wide range of learning topics. Second, the categorization is separately compared against the

classification of learning outcomes for which the experts showed a strong agreement. Tab. 8.13 summarizes results of all possible combinations of different sets of the curricula used for building the dictionary for *DL* categorization. The header denotes analyzed curricula. Results rounded to two decimal digits use the format *[All CS DL]* to show overall relative match of categorization and relative match for competencies with strong expert agreement. Best scores per column and category are marked bold. The results show the fraction of matching categorizations. E.g., row five shows the approach achieves a match with expert ratings for 74% of all learning outcomes of CSTA 11 using *BC* and *DK* for the *DL* dictionary. For the categorization of uniformly classified learning outcomes, this configuration matches in 94% of the *DL* learning outcomes, and 94% of all those learning outcomes. CSTA 11 does not contain uniformly classified *CS* learning outcomes, indicated with '—'. No single dictionary performs best for the categorization of all analyzed curricula. The best overall categorization scores are achieved by *DK* for a single curriculum dictionary, with scores in the range .64 to .75 and a mean score of .70, and by the combination of *BC* and *DK* for a multi-curriculum dictionary, with scores in the range .68 to .74 and a mean score of .70. Notably, these two dictionaries perform best for two different sets of analyzed curricula. Regarding categorization of uniformly classified learning outcomes, again no single dictionary performs best. Measured with the sum of matching categorization scores, the dictionary built with *BC* performs best for categorizing *CS* learning outcomes and overall uniformly classified learning outcomes, with mean scores of .81 and .90, respectively. The dictionary built with all three curricula performs best for categorizing *DL* learning outcomes, with a mean score of .99, mismatching one learning outcome.

Pasterk, Kesselbacher and Bollin go on by discussing their results [PKB19, p. 8]: With the help of the experts' categorization, it was possible to identify the foci of the selected educational models. For automated categorization, the best performing sets of dictionaries with an accuracy of 70% are *AIE* for *CS* and *DK* for *DL* or the combination of *DK* and *BC* for *DL*. To identify the foci of the educational models, the numbers and fractions of categorized learning outcomes are presented in Tab. 8.14. As it can be seen for the Australian curriculum (AU) the dictionary based on *AIE/DK* tends to identify a focus in *DL* (.68 compared to .32 in *CS*). Following the results of the dictionary based on *AIE/BC*, *DK* this curriculum is balanced (.41 for *DL* and .36 for *CS*). This balanced view corresponds to the results from the experts' choices. For the curriculum 21 from Switzerland a clear focus on *DL* can be identified with both dictionaries having similar results, (.77 to .80 in favor of *DL*). This result corresponds to the experts' categorization. A similar situation can be seen for the CSTA standards from 2011 where a focus for *DL* is visible (.72 in favor of *DL*). Here, again the results from the experts also indicate a focus on *DL*. Following the experts' results, the CSTA standards from 2017 tend to be balanced which is also reflected by the semi-automated generated results (.39

Tab. 8.13: Comparison of results (adapted from [PKB19])

Curricula for DL dict.	AC (AU) all Agreement			21 (CH) all Agreement			CSTA 11 (US), all Agreement			CSTA 17 (US), all Agreement		
BC			.73			.70			.65			.64
	1.0	1.0	1.0	.86	.67	.89	.88	—	.88	.86	.78	1.0
Dig			.73			.57			.67			.67
	.89	1.0	.50	.64	.67	.63	1.0	—	1.0	.71	.56	1.0
DK			.73			.75			.70			.64
	.89	.86	1.0	.91	.67	.95	.94	—	.94	.79	.78	.80
BC, Dig			.73			.64			.72			.64
	.89	1.0	.50	.73	.34	.79	1.0	—	1.0	.71	.56	1.0
BC, DK			.68			.70			.74			.69
	.89	.86	1.0	.81	.67	.84	.94	—	.94	.86	.78	1.0
Dig, DK			.73			.70			.67			.64
	.89	.86	1.0	.77	.67	.79	1.0	—	1.0	.71	.56	1.0
BC, Dig, DK			.68			.75			.72			.67
	.89	.86	1.0	.91	.67	.95	1.0	—	1.0	.79	.67	1.0

to .49 for *CS* and .44 for *DL*). Summarizing, the semi-automated categorization matches the experts' opinions in the identification of the focus for the majority of the analyzed educational models. In three cases, both dictionaries of the semi-automated approach identified the same foci as the experts did. In one case, only the results from the dictionary based on *AIE/BC*, *DK* corresponded with those of the experts. An important conclusion is that the quality of categorization is highly dependent on the curricula used for building the categorization dictionary. The semi-automated approach presented in this contribution shows a few threats to validity. At first, the translation of the learning outcomes from the German-language curriculum can lead to the use of different terms. This can result in a lower frequency of important terms. Because all of the experts were from Austria it can also be the case that they are biased by the local, well-known *digikomp* (*DK*) competency model which was chosen to build the dictionaries. This can be a factor for the good performance of the dictionary based on *DK*. Another threat can be that expert categorization can also be wrong, invalidating the comparison.

Tab. 8.14: Application of categorization with two different dictionaries (adapted from [PKB19])

		AC (AU)		21 (CH)		CSTA 11 (US)		CSTA 17 (US)	
CS	Experts	10	.45	10	.23	10	.23	20	.51
	AIE/DK	7	.32	7	.16	8	.19	19	.49
	AIE/BC,DK	8	.36	7	.16	9	.21	17	.44
DL	Experts	11	.50	32	.73	32	.75	18	.46
	AIE/DK	15	.68	35	.80	31	.72	15	.39
	AIE/BC,DK	9	.41	34	.77	31	.72	17	.44
Unde- cided	Experts	1	.05	2	.04	1	.02	1	.03
	AIE/DK	0	.00	10	.23	4	.09	5	.12
	AIE/BC,DK	5	.23	7	.16	3	.07	5	.12

Pasterk, Kesselbacher and Bollin conclude with the following words [PKB19, p. 9]: The best performing dictionaries achieve a matching categorization of 70% of all learning outcomes of the analyzed curricula. Furthermore, for learning outcomes which were uniformly classified by the experts (at most one expert disagreed), the best performing dictionary achieves a matching categorization of 90% of those learning outcomes. The results suggest that the focus of a curriculum regarding the two categories, *computer science* and *digital literacy*, can be identified with the application of the approach.

8.6.3 Five Category System

In a second and different approach the results from the text analysis in section 8.3.2 are used to categorize the competencies into the five categories for computer science education *digital applications*, *digital infrastructure*, *programming and algorithms*, *data representation*, and *human factors and ethics* [DB15]. As described in section 8.3.2 the majority of the found nouns only occur in one category. This fact is used to build *training sets* for categorization algorithms in a similar way like the dictionaries are used in the first approach (see section 8.6.2). They use a subset of the nouns occurring in the categories to automatically identify a suitable category for a competency. From each curriculum and category a subset of the competencies is randomly selected to generate the *training set* whereas the remaining competencies build the *test set*. From these competencies the keywords are extracted and stored in

a dictionary for the corresponding category. As this process works randomly it is not sure that all unique keywords for the categories are found. Manning, Raghavan, and Schütze describe text classifiers which are based on probabilistic models [MRS08]. In case of this approach only basic probabilities are calculated as it is based on the uniquely occurring keywords of the categories which are 65.8% of all nouns. The ideal case is that one keyword from a category is found in a competency which enables an automated categorization. If two or more keywords occur in one competency formulation, the decision is based on the distribution of the unique nouns over the categories (see section 8.3.2). That means, the value of how many unique nouns occur in one category is crucial and the category with the highest value of all found categories in one formulation is selected. The probability to categorize correctly is higher this way than doing it randomly. If no keyword occurs in a formulation, it cannot be categorized in this approach [Chy20].

This approach is tested with the standardized competency formulations and evaluated using the already categorized data. The three different sizes for the *training set* 80%, 67%, and 50% are tested and evaluated as well to find the best size for the training set. The *test set* is categorized following the nouns collected from the *training set* and the results are compared to the available data with the following percentages of matches:

training set 80%: 43.47% (60 out of 138)

training set 67%: 44.05% (100 out of 227)

training set 50%: 49.70% (165 out of 332)

That means using half of the competencies as *training set* provides the highest number of matches. Therefore, the results from the *training set* with size 50% are used in the following discussion. A detailed look into the results for the categories is shown in Fig. 8.8. It is visible that especially the small category *data representation* is, with 75%, often categorized according to the existing data followed by the category *digital applications* with 60%. The category *human factors and ethics* shows with 42.22% the lowest percentage of matches [Chy20].

Taking a look at the numbers in the single curricula it shows that also competencies from translated curricula have high matching rates. Fig. 8.9 presents the percentages of matches distributed over the curricula. The GI standards from Germany have with 58.33% the highest percentage of matches followed by the CSTA standards from 2011 with 58%, the curriculum from Switzerland with 57.69% and the CSTA standards from 2017 with 56.60%. The lowest percentage of matches is found in the Australian curriculum [Chy20].

The percentage of matches in the categories for each curriculum can be seen in Fig. 8.10. It shows that the category *data representation* has in three curricula

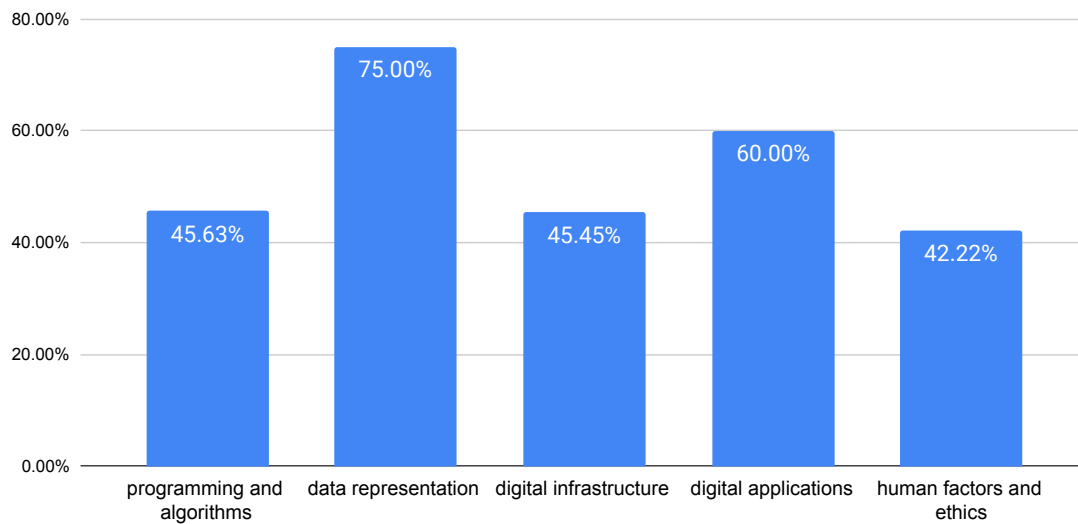


Fig. 8.8: Percentage of matches in the categories

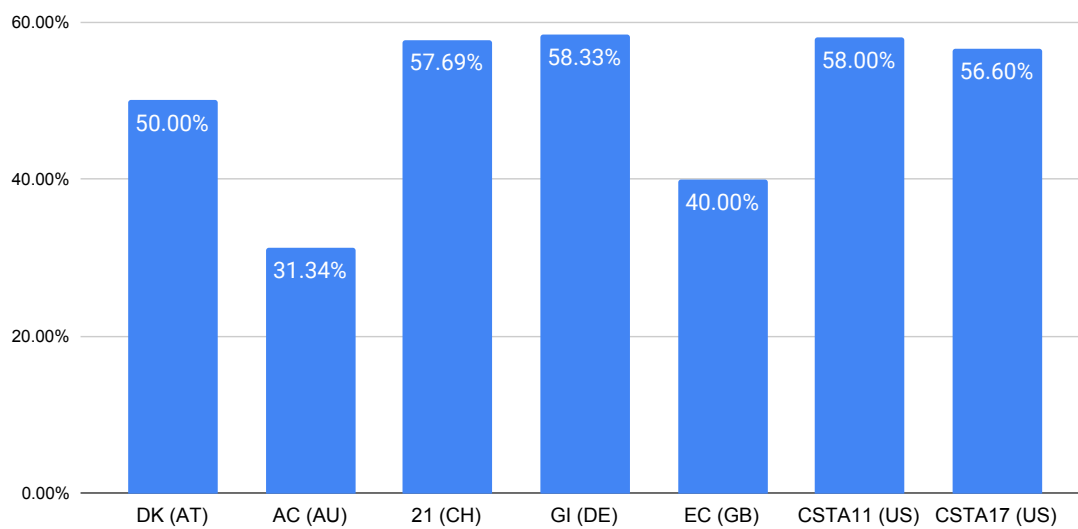


Fig. 8.9: Percentage of matches in the curricula

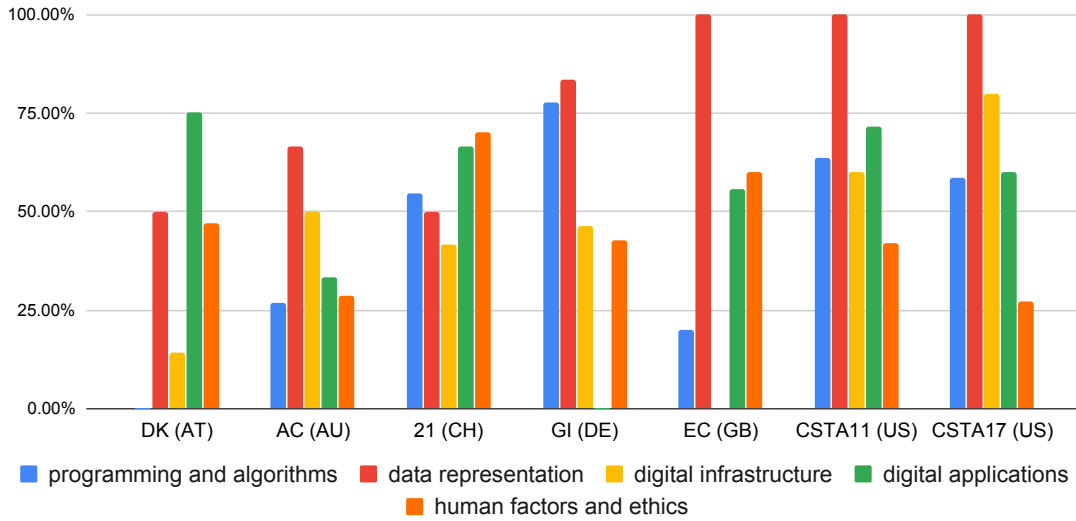


Fig. 8.10: Percentage of matches in the categories within the curricula

(EC, CSTA11, CSTA17) 100% matches in the categorization. As this category has a very low number of competencies the absolute numbers for these matches are one competency in the EC, one in the CSTA11, and three in the CSTA17. The highest percentage of matches beside the category *data representation* can be found the CSTA17 and the category *digital infrastructure* (80%), the GI and the category *programming and algorithms* (77.78%), and the DK and the category *digital applications* (75%). For the DK and the category *programming and algorithms* and for the GI and the category *digital applications* no matches are found. In the DK and the category *digital infrastructure* the lowest percentage of matches with 14.29% is visible, followed by the EC and the category *programming and algorithms* with 20% [Chy20].

The results of this approach show that the categorization using keywords from a *training set* of the curricula works in more or less 50% of the cases. Although, it has to be mentioned that the size of the used data set is very small for approaches which use training and test sets. Compared to the categorization into *computer science* and *digital literacy* from section 8.6.2 this result is a bit lower. But it has to be considered that the first approach had a chance of 0.5 to randomly hit the right category as there are only two categories. In the second approach the probability to randomly find the same category as in the existing data is only 0.2 as there are five categories. Therefore, the results for both approaches are promising.

8.7 Automated Combination of Competencies

Comparable to the approach to generate the relations between the competencies (see section 8.5) the idea to identify competencies for *intersector nodes* is also based on linguistic similarity of the competency formulations. In contrast to the process to identify the relations, which is limited to the single curricula, for the combination to *intersector nodes* the similar competencies in all curricula have to be found. It has to be considered that the possible combinations of competencies are not, like in case of the relations, limited to single curricula. This means the number of competencies one competency can be combined with is for a graph G with n nodes again $n * (n - 1)$. But, as all curricula are considered, n is 688 and with that there are 472,656 possible combinations. To find those competencies, which have a level of similarity that allows a combination, again the *Jaccard coefficient* is used. Relations between subcompetencies from the same competency cluster are excluded as they, in most cases, only differ in very few words.

In this approach pairs of similar competencies are found which can, based on linguistic features, be combined to *intersector nodes*. The results from this process are again compared to the results from the process done for this dissertation and described in section 4.6.4 and 6.2.2.

As described in section 6.2.2, 238 competencies are manually combined to 88 *intersector nodes*. But these values do not represent the possible combinations of competency couples. For the 88 *intersector nodes* and the combined competencies in this case 267 possible combinations of couples exist. These are calculated using the *binomial coefficient* and the number of combined competencies for each *intersector node*. The *binomial coefficient* is defined as follows [Bru10, p. 42, 128]:

Definition 8.7 (Binomial coefficient). For $0 \leq k \leq n$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Where,

n is the number of elements in a set, and

k is the number of elements chosen from the set.

Using the numbers of combined competencies for each *intersector nodes* as the number of elements in this set and two as the number of elements to be chosen, the number of possible pairs in each *intersector node* can be calculated. Summing up all these numbers up 267 possible combinations to pairs within the *intersector nodes* can be found.

The automated approach delivers the following numbers of possible pairs depending on the used threshold for the *Jaccard coefficient*:

Jaccard coefficient 0.7: 8 pairs

Jaccard coefficient 0.6: 37 pairs

Jaccard coefficient 0.5: 165 pairs

Jaccard coefficient 0.4: 227 pairs

These numbers are comparable to the possible combinations from the *intersector nodes*. But taking a look at how many of the automatically found pairs match with those determined manually, only a small number of exact matches is visible.

Jaccard coefficient 0.7: 3 matches

Jaccard coefficient 0.6: 3 matches

Jaccard coefficient 0.5: 5 matches

Jaccard coefficient 0.4: 3 matches

A detailed analysis of the automatically found pairs shows that 95 of them are closely related but not close enough to be combined to *intersector nodes*. 118 found pairs are evaluated as being not related at all.

This means that of 88 *intersector nodes* only 2 are completely and 11 partly found by linguistic similarities. These results lead to the assumption that the combination to *intersector nodes* is not based on linguistic features like keywords but on experts' opinions and experience. It has to be shown in additional tests and future work if this approach to semi-automated combined competencies can be improved.

8.8 Conclusion

This chapter presents different approaches to semi-automated processes done as preparation for this dissertation. All of them are based on *linguistic features* and *natural language processing*. The main processes, that are supported by this automation, are the generation of relations between competencies within each curriculum, the categorization of the competencies, and the combination of very similar competencies in the different curricula to *intersector nodes*. Methods from *natural language processing* and well known libraries within this field of research are found and support this approach. The text-based analysis shows that the keywords in a curriculum can give information about the focus. From the perspective of part of speech the *nouns* are highly decisive and informative. *Verbs* add information about the type of activities fostered by a curriculum. *Adjectives* and *adverbs* can be used to gather more detailed information about the focus. Taking a look at the occurring

words in the categories for computer science education, *digital applications*, *digital infrastructure*, *programming and algorithms*, *data representation*, and *human factors and ethics* [DB15], it can be seen that 65.8% of the nouns only occur in one category and only 1.1% of the nouns occur in all five categories.

To support the generation of relations between the competencies in one curriculum, *similarity measures* are used. The results of different methods are compared by calculating *precision* and *recall*. In a first step it is tested which combination of part of speech delivers the most results which accord to those generated by experts. The best performing combination is, in this case, the combination of *nouns* and *verbs*. Additionally, the results of two different similarity measures, the *cosine similarity* and the *Jaccard coefficient*, are evaluated in their accordance to the existing data. It turned out that the *Jaccard coefficient* shows higher values in both *precision* and *recall*. That is why this method, together with the part of speech combination *nouns* and *verbs*, is used for further research. The results could even be improved by adding weights and synonym replacement into the preprocessing. Overall, a value of 0.26 for *precision* and 0.44 for *recall* are reached. As the *recall* value is more important in this case, these results are promising. But it turned out that only *expands* relations are identified by this method. A different approach will be developed in future work to also identify *requires* relations.

The automated categorization is tested in two different experiments. In a first approach dictionaries are generated using curricula which are not part of the analysis to categorize into *computer science* or *digital literacy*. For the category *computer science* the *ACM/IEEE curriculum for computer science (AIE)* [JTFoCCS13b] and for *digital literacy* the *The Digital Competence Framework for Citizens (Dig)* [CVP17], the *British Columbia Digital Literacy Framework (BC)* [oBC13], and the *digikomp4 (DK)* [Dig13b] are selected. The results are again compared to the categorization of experts (see section 5.5) and those dictionaries with the most matches are seen as best performing. It turned out that the best performing combination are the dictionaries from *AIE* for *computer science* and from *DK* for *digital literacy*. In some cases a combined dictionary of *DK* and *BC* for *digital literacy* delivers even better results. Overall 70% of the competencies could be categorized according to the opinions of the experts. Taking into account that the experts only agree in a part of the categorizations, 90% of the categorizations with strong agreements are found. The second approach, concerning the automated categorization, is based on the already categorized data and a *training set* from this data. Using this *training set* for the five categories for computer science education *digital applications*, *digital infrastructure*, *programming and algorithms*, *data representation*, and *human factors and ethics* dictionaries are generated. Based on the results from the analysis of term occurrences in the categories, keywords in form of *nouns* are collected for all categories and used to evaluate the approach in a *test set*. The results from the evaluation show that overall 49% of the competencies are categorized according to

the existing data. This is also a good result considering the fact that in this case five possibilities for categorization are given. In future work this approach could possibly be improved by using probabilistic models.

To support the combination of competencies to *intersector nodes* from all curricula, an approach based on linguistic similarities is developed and tested. Again the similarity measure used is the *Jaccard coefficient* which delivered good results in previous research. Compared to existing data only 13 *intersector nodes* out of 88 are partially identified. This leads to the assumption that the combination to *intersector nodes* is not based on linguistic features. Future work will show if different approaches, possibly based on ontologies, could improve these results.

9. PROJECT 'LAKESIDE IT-CURRICULUM'

9.1 Introduction and Description of the Project

As a use case of the approach of this dissertation the project called *Lakeside IT-Curriculum* is started as a collaboration between *Lakeside Science and Technology Park*¹ and the *University of Klagenfurt*². The *Lakeside Science and Technology Park* can be seen as a linking point between research, enterprises, and education and runs its own *Educational Lab*³ which can be described as follows:

The Educational Lab is an open research laboratory for new forms of education and further and continued training and development.

The project *Lakeside IT-Curriculum* has the goal to develop a continuous curriculum for computer science and digital literacy starting in kindergarten and continuing until graduation. With this idea the project won the first price in the competitive *Inspiring Solutions Programme*⁴ from the *International Association of Science Parks and Areas of Innovation (IASP)*⁵.

It is divided into the three following phases:

1. **Kindergarten and first years of primary school:** in the first phase the lowest levels are covered. This marks a special challenge as reading and calculation skills are very limited and cannot be seen as prerequisite skills. This has to be considered during the development of activities and material.
2. **Primary school and lower secondary school:** in higher grades of primary education reading and calculations skills are on a basic level and can be presumed in the activities and material.
3. **Secondary school:** in higher grades of secondary school a lot more knowledge and skills can be expected. This means, some topics can be additionally considered or more details can be integrated.

¹ <https://www.lakeside-scitec.com/en/>

² <https://www.aau.at/>

³ <https://www.lakeside-scitec.com/en/educational-lab/educational-lab/>

⁴ <https://www.iaspinspiringsolutions.com/2019-winners>

⁵ <https://www.iasp.ws/>

As the project is not part of the official curriculum for these school grades, but are implemented in the *Educational Lab*, the activities are planned as workshop modules. In the first pilot study, together with the public kindergarten *Bunte Knöpfe (Coloured Buttons)*⁶ in Klagenfurt, Carinthia, Austria, the first phase is implemented. Some first results from this first phase are part of this dissertation as far as they are connected to its approach. The project includes several aspects of development which cover different components of a curriculum. These steps can be summarized as follows:

1. **Formulation of learning outcomes:** competencies, which are suitable for this age and are relevant for the project, are collected and reformulated.
2. **Definition of a sequence:** during this step the sequence of competencies is determined.
3. **Planning and documentation of activities:** based on the selected competencies and their sequence the activities are planned and documented using a given template which includes all necessary information for educators and teachers.
4. **Creation of material for the activities:** as the activities often need additional material this has to be prepared in digital or also tangible form. This material is included in the documentation of the activities and can again be reused by educators and teachers.
5. **Testing of the activities:** during a pilot phase the planned and described activities are carried out in kindergarten, respectively in primary school classes. In this step it is important to document the behavior of the children and their reaction on the presented material.
6. **Revision of documentation and materials:** based on the observations from the testing step the material and documentation of the activities has to be revised and finalized.

For the basic steps 1 and 2 the graph-based approach presented in previous chapters of this dissertation is used to identify required competencies and the sequence they are to be taught. Therefore, steps 1 and 2 are described in more detail in the following section 9.2. All results in form of sets of competency formulations (which are collected to modules) are presented in section 9.2.3. Section 9.3 presents results from an questionnaire for the educators involved in this pilot phase and their opinions concerning the components of the curriculum relevant for this dissertation.

⁶ <https://www.klagenfurt.at/leben-in-klagenfurt/kindergaerten-horte/kindergaerten/bunte-knoepfe.html>

As the project is an use case for the graph-based approach, its results are used to evaluate the automated approach in the *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*⁷ (see chapter 7) to generate learning paths. In section 9.4 the results from automated determination of learning paths are compared to those selected for the project to evaluate the methods and metrics.

9.2 Determination of Competencies and Learning Paths

9.2.1 Selection of Competencies

To prepare the development of the curriculum for the first phase, so kindergarten and lower primary school, targeted competencies from the *GGBMC* had to be selected. The selection of these target competencies is based on discussions between experts for computer science education from the *Department of Informatics Didactics*⁸ from the University of Klagenfurt, educators from the kindergarten *Bunte Knöpfe (Coloured Buttons)*, and the team of the *Lakeside Science and Technology Park*. Taking the ecosystem of the *Lakeside Science and Technology Park* and the *Educational Lab* into consideration, the following four points of interest were found:

- Computational Thinking
- Software Development
- Engineering Abilities
- Hardware Competences

These very general points form the basis for further development. An additional limit is of course the age group of kindergarten and lower secondary education which starts in Austria at an age of 3 years and lasts till the children are 8 years old. This means, only competencies with lowest minimum age levels are valid. Also, the project is restricted in terms of time due to demands of the kindergarten. Because it is limited to half a year, not all found competencies or learning paths are included.

To visualize the competencies for the discussions, they are collected in a table as the graph representation is limited in getting an overview of the competency formulations. The table is structured in the curricula (columns) and each competency is added in an own row. Similar competencies are collected in the same row to also get an overview of intersections. Fig. 9.1 shows all the columns of the table with an example of competencies in the category *digital applications*. Cells in blue show original competency formulations for level 1 and green cells for level 2. The

⁷ <https://gecko.aau.at>

⁸ <https://www.aau.at/en/informatics-didactics/>

orange cells are subcompetencies from the original formulations. With the yellow and red cells above, the relations of the subcompetencies are displayed. These cells include the internal IDs of the competencies or subcompetencies which the corresponding subcompetency is related to by *expands* (yellow) and *requires* (red). A

CSTA 2011 (US)		CSTA 2017 (US)		EC (GB)		International Curricula		digikomp (AT)		GI (DE)		21 (CH)	
ID	Competency	ID	Competency	ID	Competency	ID	Competency	ID	Competency	ID	Competency	ID	Competency
						Digital Applications							
		705	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.	204	Use technology purposefully to create, organize, store, manipulate and retrieve digital content.	300	collect, sort and display familiar data from a range of sources and recognize patterns in data.	426	Ich kann Netzwerke zum Suchen und Darstellen von Informationen nutzen.				
		705.1	requires			302	expands	426.1	requires				
		705.2	expands	204.2	expands	304.2	expands	426.2	expands				
		705.3	Search information using a computing device.	204.1	I can use technology purposefully to retrieve digital content.	305	collect familiar data from a range of sources	426.1	Ich kann Netzwerke zum Suchen von Informationen nutzen.				
								426.2	expands				
						300	collect, sort and display familiar data from a range of sources and recognize patterns in data.						
						305.1	expands						
						305.2	sort familiar data from a range of sources						
		706	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.	204	Use technology purposefully to create, organize, store, manipulate and retrieve digital content.	300	create and organize ideas and information using information systems, and share information in safe online environments.	440	Ich kann Daten erfassen, speichern und ändern.	534	Die Schülerinnen und Schüler speichern Daten und finden Daten wieder.		
		706.1	requires										
		706.2	Store information using a computing device.	204.2	I can use technology purposefully to store digital content.					535	expands		
		706.3	expands	204.2	expands	305.2	expands	440.1	expands	536.2	expands		
		706.4	Copy information using a computing device.	204.3	I can use technology purposefully to organize digital content.	305.3	organize ideas using information systems	440.2	Ich kann Daten speichern.	534.1	Die Schülerinnen und Schüler speichern Daten.		
		706.5	expands			305.3	expands			534.1	expands		
		706.6	Retrieve information using a computing device.			305.4	organize information using information systems			534.2	Die Schülerinnen und Schüler finden Daten wieder.		
		706.7	expands	204.4	expands			440.2	expands				
		706.8	Modify information using a computing device.	204.5	I can use technology purposefully to manipulate digital content.			440.3	Ich kann Daten ändern.				
		706.9	expands										
		706.10	Delete information using a computing device.										
		706.11	expands										
		706.12	Define the information stored as data.										

Fig. 9.1: Table representation of all competencies for discussion.

readable extract of this table is shown in Fig. 9.2. Here, three similar competencies from the CSTA standards from 2017, the model for the English curriculum, and the Australian curriculum are put in the same row to represent their combination to an *intersector node*.

Working on these tables (and based on the identified limitations) the following *target competencies* are defined for the project:

- Computational Thinking:

INT-005-PA: The students are able to use information technology to capture ideas and stories in a step-by-step manner.

INT-003-DR: The students are able to recognize patterns in data.

US1-11.0: The students are able to construct a set of statements to be acted out to accomplish a simple task [SCF⁺11].

- Software Development:

US1-6.0: The students are able to recognize that software is created to control computer operations [SCF⁺11].

International Curricula								
CSTA 2017 (US)			EC (GB)			AC (AU)		
ID	Competency		ID	Competency		ID	Competency	
Digital Applications								
705	Store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.		204	Use technology purposefully to create, organise, store, manipulate and retrieve digital content		309	collect, sort and display familiar data from a range of sources and recognise patterns in data.	
	701.2 requires			204.2 expands			302 expands 304.2 expands	
	705.3	Search information using a computing device.		204.1	I can use technology purposefully to retrieve digital content		309.1	collect familiar data from a range of sources

Fig. 9.2: Table extract showing three similar competencies.

GB-2.2: The students are able to debug simple programs [Ber15].

- Engineering Abilities:

AT-48.0 The students are able to create simple instructions [Dig13b].

CH-33.2: The students are able to test the correctness of solutions for simple problems by trial and error [Leh14].

- Hardware competences:

CH-6.2: The students are able to experiment creatively with software to create digital content [Leh14].

INT-002-HF: The students are able to talk about their own use of information technology.

It has to be mentioned that one selected competency for *Engineering Abilities* from the curriculum from Switzerland shows a higher minimum age level: 'The students are able to test the correctness of solutions for simple problems by trial and error' [Leh14]. But, during the discussions, this competency was seen as essential and should be fostered as soon as possible.

These *target competencies* are in most cases no *sinks* and show some dependencies to other competencies. So, learning paths, as described in section 6.4, are generated to determine the prerequisites for the *target competencies*. This process is described in the following section.

9.2.2 Transforming generic Learning Paths to the Project

The selected *target competencies* build the basis for the determination of learning paths for the project and, with that, necessary competencies which are no *target competencies*. For all of them learning paths to starting points, competencies which do not depend on others, are manually generated and discussed in the project-team, mainly because of the connection of *intersector nodes*, for most *target competencies* more than one path is found. So, the learning paths to include in the project are selected again during discussion in the project-team. Additionally, it turns out that to enable some activities, further competencies are necessary especially in the handling of media and the digital devices. For this purpose, the following five so called *support competencies*, which focus on the use of digital media and devices, are selected by the project-team:

CH-2.2: The students are able to talk about simple contributions in different media languages [Leh14].

AT-35.2: The students are able to use digital video files [Dig13b].

INT-001-DA: The students are able to create digital products with guidance.

INT-001-HF: The students are able to name digital devices in their daily environment.

AU-6.2: The students describe how familiar products, services and environments meet a range of current needs [Aus13].

To visualize the results and again to enable collaboration and discussion, the table shown in Fig. 9.1 and Fig. 9.2 is used. It is reduced to those competencies which meet the criteria mentioned in the previous section. Additionally, the cells are ordered in a tree-like form to display dependencies and learning paths. This results in the representation shown in Fig. 9.3. Again, the competencies are represented by their formulations in blue and green cells. The *target* and *support competencies* can be found on the bottom, whereas the competencies to begin with are on the top of the diagram. As this diagram gives only an overview of the dependency structure for the competencies of this project, the following three more detailed examples are extracted from it and discussed. The color codes of the diagram are also explained in this process.

A very straight forward example of a dependency structure can be found in Fig. 9.4. Here, a different form of *identification numbers (ID)* for the competencies is applied. As it should not be indicated from which curriculum a competency comes from, the numbering is connected to the corresponding categories. In the legend next to the diagram the numbering for the categories is specified. At the bottom it shows

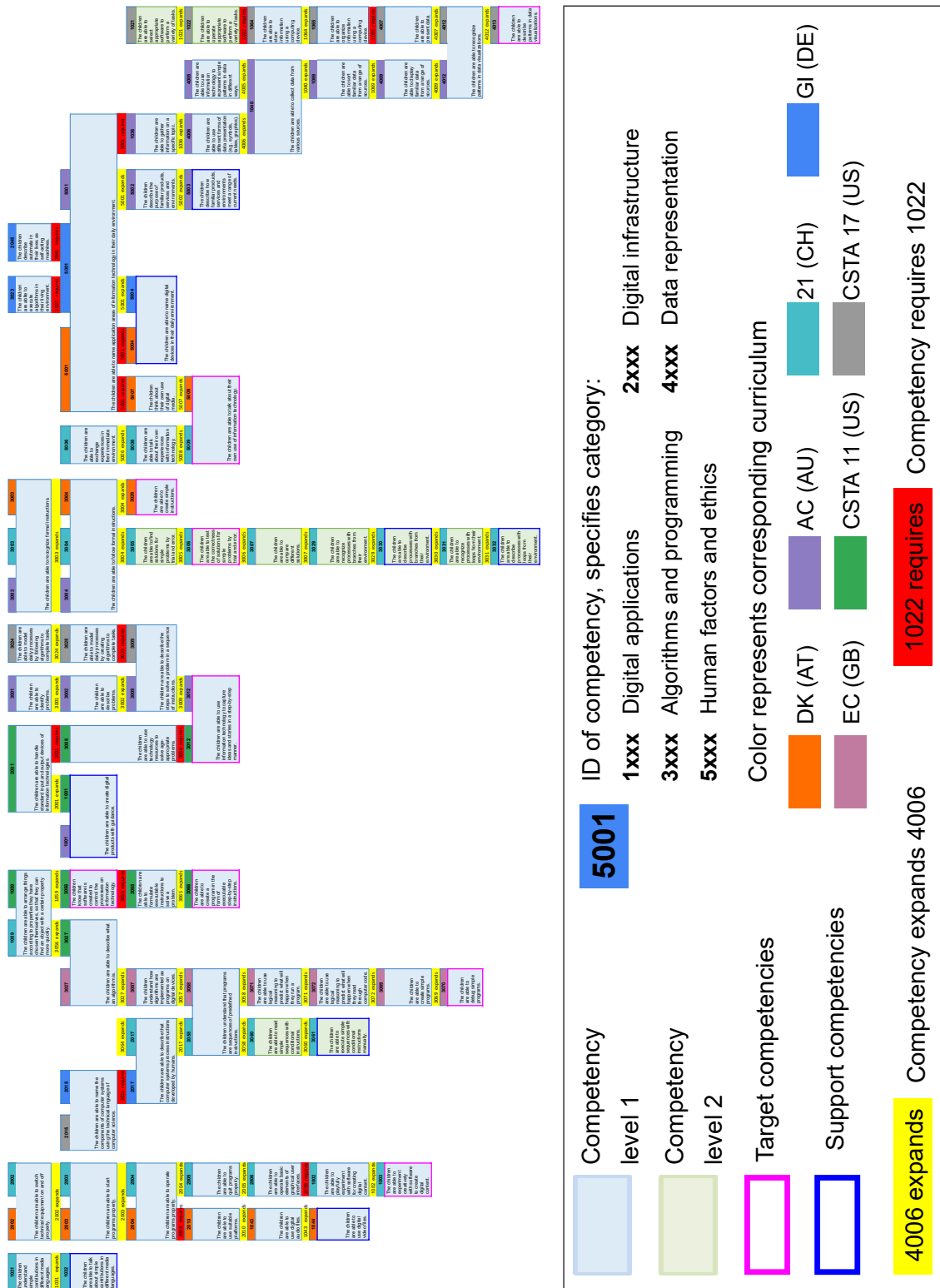


Fig. 9.3: Dependencies structure for all selected competencies

the *target competency* 1003 (with borders in magenta) and the *support competency* 1044 (with blue borders). The competencies these two depend on follow upwards in the diagram. So, competency 1003 depends on competency 1002 in form of an *expands* relation (shown by the yellow cell). The colors of the cells including the competency ID reflects the curriculum. This has the advantage that paths from different curricula can be differentiated. In case of Fig. 9.4 these are the Austrian competency model and the curriculum from Switzerland. Both paths depend on the competency 2004 which is an *intersector node*. From this point of combination on they follow the same path to a competency which does not depend on other competencies.

In case of Fig. 9.5 a bit more complex example of dependency structures is shown. The *target competency* with the ID 3012 is already a combination of competencies from two curricula. This means that at least two paths can be selected. The *support competency* 1001 is also a combination and has on one side no dependencies but *expands* on the other side competency 2001, which is again a combination without any dependencies. For the project two paths are selected as both contain important competencies. One of them is the path in green to the competency 2001 and represents competencies from the CSTA standards from 2011. The other one is the gray path to the competency 3024 and derives from the CSTA standards from 2017.

Fig. 9.6 shows a very complex case of dependency structure because the competencies are very interconnected. It includes the two *target competencies* 4013 and 5009, and the two *support competencies* 5003 and 5004. For both *target competencies* different paths to competencies, which are identified as starting points, are available. The project follows, in case of competency 4013, the purple path passing competency 1040 and ending at competency 5001. As illustrated in the legend of Fig. 9.6 the purple competencies belong to the Australian curriculum. For *target competency* 5009 the orange path to competency 5001 is considered which derives from the Austrian competency model. Although competency 5001 has no additional dependencies in the Austrian and Australian curriculum, competency 3023 is also included as it is an important competency from the GI standards. The two *support competencies* do not need any decision as both only have one path to competency 5001 which acts as linking point in this example.

With these three representative examples the process of transforming the learning paths into the learning outcomes of a curriculum are described. In the following section extracts of the resulting curriculum are presented and discussed.

9.2.3 Resulting Curriculum

The preliminary work and analysis, as well as the discussions, lead to the basis of the curriculum for the kindergarten-project. As already mentioned the learning outcomes are grouped in modules to cover them in workshops besides the regular

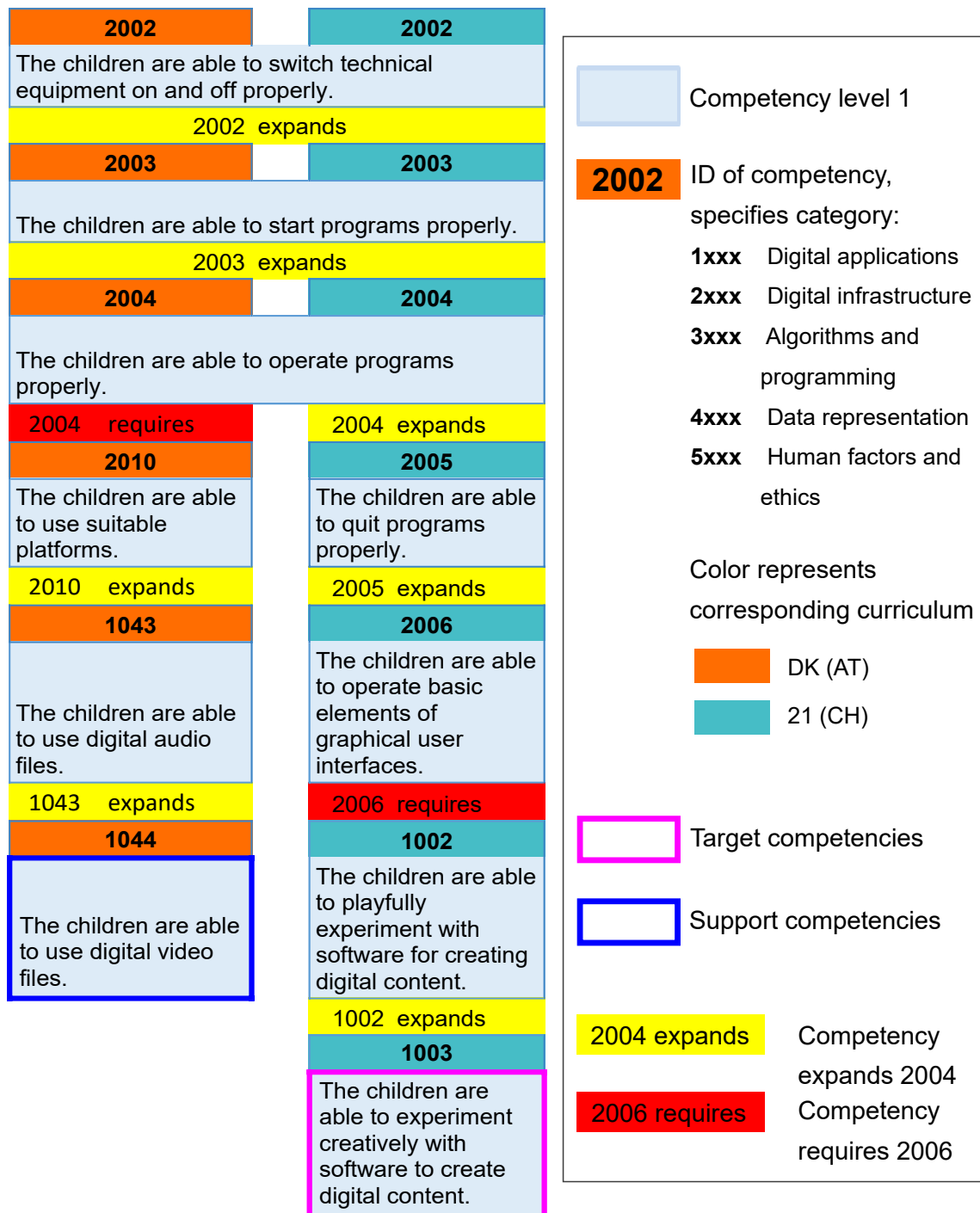


Fig. 9.4: Simple dependency structure for one *target competency* and one *support competency*

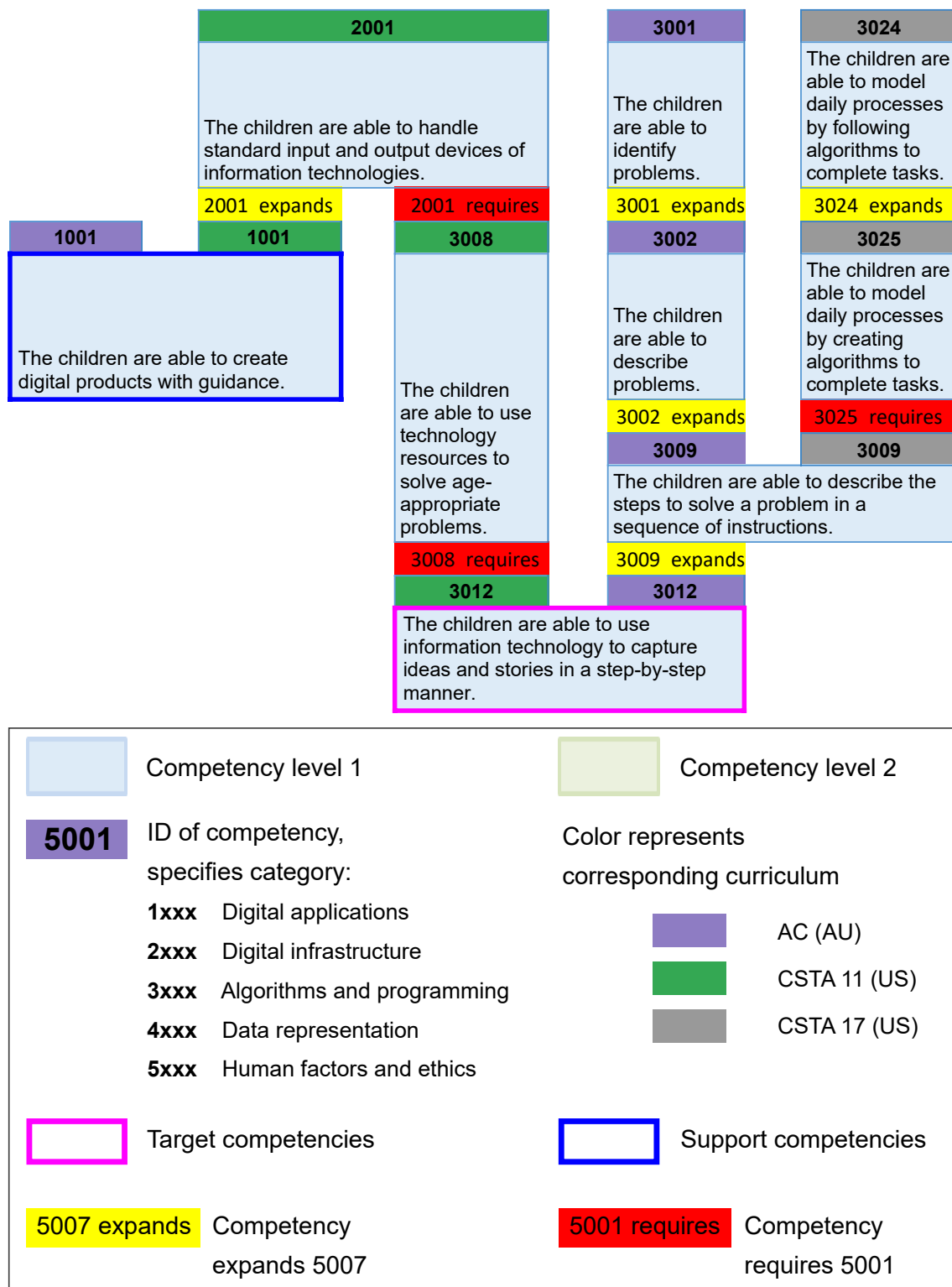
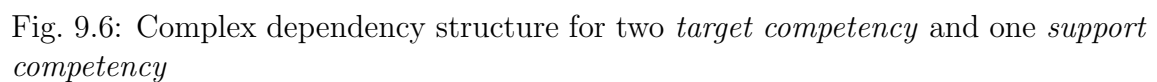


Fig. 9.5: More complex dependency structure for one *target competency* and one *support competency*



school activities. For the project the following ten modules are defined:

0. Use of technology
1. Technology in the daily life (includes parts from module 0)
2. Understand digital media (includes parts from module 0)
3. Step-by-step instructions (includes parts from module 0)
4. Problem solving
5. Algorithms in the daily life (requires module 3, 4)
6. Characteristics and patterns (requires module 1)
7. Programming 1
8. Programming 2 (requires module 1, 6, 7)
9. Programming 3 (requires module 8)

Some of the modules depend on each other which requires a sequential ordering. Module 0 *use of technology* is a special case as in the project it is not intended to cover its competencies in own workshops. The children learn to handle the devices in the workshops in which they are necessary. This module includes the following competencies:

0. Use of technology

1001: The children are able to create digital products with guidance.

1002: The children are able to playfully experiment with software for creating digital content.

1003: The children are able to experiment creatively with software to create digital content.

2001: The children are able to handle standard input and output devices of information technologies.

2002: The children are able to switch technical equipment on and off properly.

2003: The children are able to start programs properly.

2004: The children are able to operate programs properly.

2005: The children are able to quit programs properly.

2006: The children are able to operate basic elements of graphical user interfaces.

2010: The children are able to use suitable platforms.

3008: The children are able to use technology resources to solve age-appropriate problems.

So, in a workshop, the children can participate in activities which often require digital devices and if they need help with the used devices they can ask educators and workshop instructors for help. It is assumed that the children learn to use the devices and reach the competencies during these activities in the workshops.

The modules *technology in the daily life* and *step-by-step instructions* are basic modules with very few prerequisites. Therefore, these two are discussed in this section. As the first module *technology in the daily life* includes a lot of activities to understand and realize the pervasiveness of technology, it is one of the most basic modules. It includes the following competencies:

1. Technology in the daily life

5001: The children can name application areas of information technology in their daily environment.

5002: The children describe the purpose of familiar products, services and environments.

5003: The children describe how familiar products, services and environments meet a range of current needs.

5004: The children can name digital devices in their daily environment.

5007: The children think about their own use of digital media.

5009: The children can talk about their own use of information technology.

2015: The children are able to name the components of computer systems using the technical language of computer science.

As it can be seen in the list above, the competencies of module 1 cover some self-reflecting activities as well as some activities connected to observation of their environment. These competencies are reached by activities like taking pictures in their daily life using digital devices and discussing their results. Because digital devices are used, parts of module 0 *use of technology* are included in this module. Additionally, games like e.g. a *hardware memory* are used to talk about the components of digital systems.

The second module *step-by-step instructions* covers first steps for important topics like algorithms and programming. It consists of the following four competencies:

3. Step-by-step instructions

3003: The children are able to recognize formal instructions.

3004: The children are able to follow formal instructions.

3012: The children are able to use information technology to capture ideas and stories in a step-by-step manner.

3026: The children are able to create simple instructions.

The competencies from this module focus on the recognition, understanding, and creation of instructions. As the competency 3012 requires abilities in terms of the use of information technology, this module includes parts from module 0 *use of technology*. The activities to reach these competencies are to follow instructions to craft own robot helmets or to rebuild a given construction of building blocks. To create own instructions, again digital devices like cameras or smartphones are used.

9.3 Opinions from Educators

At the end of the project the educators from the kindergarten involved in the project participated in a small survey concerning their opinion on the developed curriculum. The five participating educators all are female and have worked in elementary education for 10 to 29 years. All of them have experience with workshops related to technology and four of them already developed their own material in this area. In case of workshops related to computer science, four of them have experience and three developed activities for this area on their own. Four questions concerning the educators' opinions about the curriculum offer options in form of a five point Likert scale. The results for these questions are displayed in Tab. 9.1 including the numbers of answers for the five options.

The question 'How would you assess the applicability of the curriculum for the elementary level?' is answered by two educators with 'yes, very applicable' and three with 'yes, rather applicable'. Following the opinions of the educators the order of the competencies is good but can be improved. This can be assumed from the answers to the question 'Is the order of competencies appropriate?'. Two educators answered with 'yes, very appropriate', two with 'yes, rather appropriate' and one with 'undecided'. The majority of the educators think that in the curriculum the most important topics of computer science and digital education are covered. The question 'In your opinion, are the most important topics of computer science or digital education covered by the curriculum?' is answered by four educators with 'yes' and one with 'rather yes'. On the question which topics are missing, no answers were given. Following the opinions of these five educators, the competencies in the developed curriculum are suitable for kindergarten. Hence, four of them answered

Tab. 9.1: The results from the survey with kindergarten educators.

	yes	rather yes	unde- cided	rather no	no
How would you assess the applicability of the curriculum for the elementary level?	2	3	0	0	0
Is the order of competencies appropriate?	2	2	1	0	0
In your opinion, are the most important topics of computer science or digital education covered by the curriculum?	4	1	0	0	0
In your opinion, are the competences contained in the curriculum learnable in kindergarten?	4	1	0	0	0

the question 'In your opinion, are the competences contained in the curriculum learnable in kindergarten?' with 'yes' and one with 'rather yes'.

The question for inappropriate competencies in the curriculum was answered with 'none' by one educator. The following competencies 3071 and 5004 are mentioned by one educator each, whereas competency 2015 is mentioned by two educators with the added reasoning:

2015: The children are able to name the components of computer systems using the technical language of computer science.

reason: missing necessity, children often find own names for things in their environment

3071: The children are able to use logical reasoning to predict what will happen when they run a program.

reason: questionable if possible for children in this age

5004: The children can name digital devices in their daily environment.

reason: discussion in group too long for young children, talk with children individually

These answers give important information how the curriculum can be improved in future work. Especially, the reasoning for competency 3071 is of great interest. In a revision of the workshop also the explanations, why competencies 2015 and 5004 are mentioned, will be considered.

For the project it is also important to know which competencies are most appropriate in the opinion of the educators. Especially competencies from the two modules *technology in the daily life* and *understand digital media* were mentioned in this case. Not all nominations are explained by the educators but added reasons are included in the following list.

1. Technology in the daily life

5001: The children can name application areas of information technology in their daily environment.

5002: The children describe the purpose of familiar products, services and environments.

reason: great interest from the children

5003: The children describe how familiar products, services and environments meet a range of current needs.

reason: this competency is fostered in kindergarten as preparation for school in form of photographic projects and is therefore very appropriate

5004: The children can name digital devices in their daily environment.

5007: The children think about their own use of digital media.

5009: The children can talk about their own use of information technology.

reason: this competency is fostered in kindergarten as preparation for school in form of photographic projects and is therefore very appropriate

reason: great interest from the children

2. Understand digital media

1031: The children understand simple contributions in different media languages.

reason: children should use media critically and self-determined for themselves

reason: great interest from the children

1032: The children are able to talk about simple contributions in different media languages.

reason: children should use media critically and self-determined for themselves

reason: great interest from the children

3. Step-by-step instructions

3003: The children are able to recognize formal instructions.

reason: this competency is fostered in kindergarten as preparation for school in form of building block instructions and is therefore very appropriate

6. Characteristics and patterns

4012: The children are able to recognize patterns in data visualizations.

reason: this competency is fostered in kindergarten as preparation for school and is therefore very appropriate

4013: The children are able to describe patterns in data visualizations.

reason: this competency is fostered in kindergarten as preparation for school and is therefore very appropriate

Competencies 5009 and 1031 were mentioned by four educators and with that appropriate competencies from the curriculum according to the majority of the educators. Three educators mentioned competencies 5002 and 5003. The competencies 5001, 5004, 5007, and 1032 were appropriate for two educators. Competencies 3003, 4012, and 4013 were mentioned once.

9.4 Comparison to (Semi-)Automated Results

The resulting curriculum from the kindergarten project can be seen as a manual application and use case of the graph-based approach developed for this dissertation and described in chapter 4, 5, and 6. The decisions and selections are based on the opinions of and the discussions between experts in the field of computer science education and educators. In chapter 7 the *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*⁹ is presented which offers the possibility to automatically generate learning paths for *target competencies* based on different metrics like length of the path or centrality values of the covered nodes

⁹ <https://gecko.aau.at>

(see section 6.4). The results from the kindergarten project offer the opportunity for a small evaluation of the used metrics and their accordance to the experts selections.

For that reason the dependencies of the selected *target* and *support competencies* are retransformed into the graph-representation and displayed as learning paths within the *Generic, Graph-Based Model for Competencies (GGBMC)*. That means for each *target*, respectively *support competency*, one learning path is generated. With nine *target competencies* and five *support competencies* these are 14 learning paths. For each *target*, respectively *support competency*, also automated learning paths are determined using the following methods and metrics (see section 7.4.3):

- **Shortest path**
- **Covering Central Nodes:** betweenness centrality (BC), PageRank (PR), out-degree (OD), in-degree (ID)
- **Highest Overall Centrality:** betweenness centrality (BC), PageRank (PR), out-degree (OD), in-degree (ID)

The results of the automated determination in the *GECKO* platform are compared to those which the experts selected for the kindergarten project. Fig. 9.7 shows the numbers of matches between automatically determined and from experts selected learning paths for all methods and metrics. In blue, the numbers of *exact matches* can be found representing automatically determined learning paths which cover exactly the same competencies as those selected by experts do. The methods for the automated determination sometimes deliver more than one learning path as their values are the same. The yellow fields in Fig. 9.7 show cases with more than one determined learning path which includes the learning path selected by the experts. The red numbers represent cases where one or more learning paths are automatically determined but those selected by the experts are not included at all. The results show that the combination of *covering central nodes* with the metric *PageRank (PR)* has, with 11, the most *exact matches*. This result can be explained by taking a look at the project's goals. For the early stages of education the most basic competencies are relevant, on which central competencies in higher levels depend. With the combination of *covering central nodes* and *PageRank (PR)* exactly these most basic competencies are considered for the learning paths, as it is described in section 5.4.3. Together with the combination *highest overall centrality* and *PageRank (PR)* it shows the most overall matches that also consider *included results*. The lowest number of *exact matches* shows the method *covering central nodes* with the metric *out-degree (OD)* which also has the highest number of *results included*. The lowest number of overall matches, including *exact matches* and *results included*, share the methods *shortest path*, *covering central nodes* with *betweenness centrality (BC)*, and *highest overall centrality* with *betweenness centrality (BC)* and

out-degree (OD). It has to be mentioned that in half of the cases only one path leads from a *target*, respectively *support competency*, to a competency without dependencies. This means all methods and metrics provide the one possible path and are considered as a match.

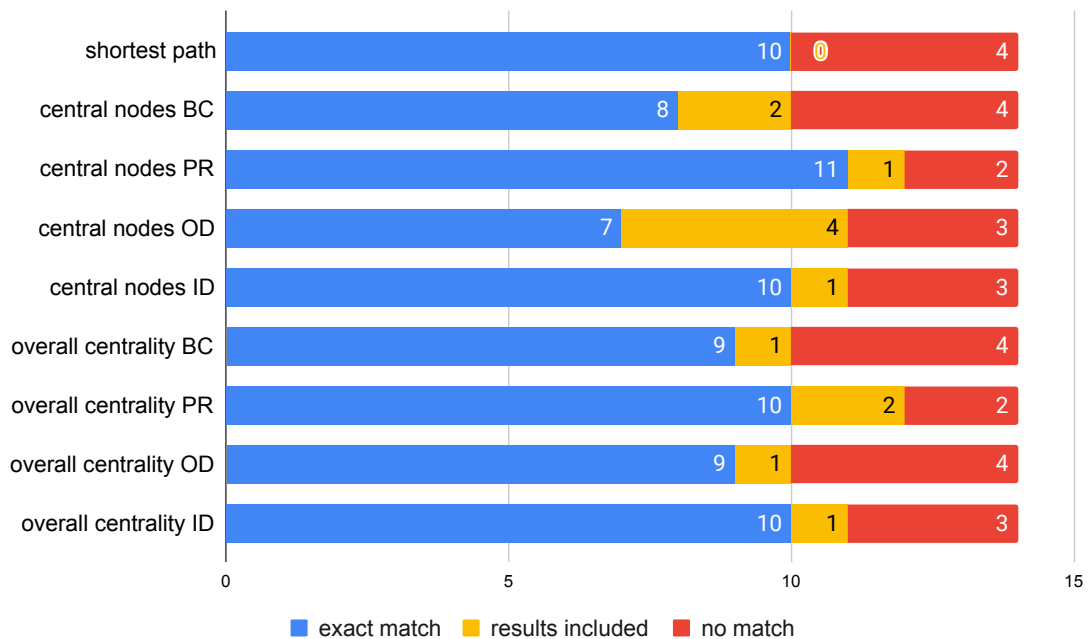


Fig. 9.7: Matches of automatically determined and from experts selected learning paths

9.5 Conclusion

This chapter presented a use case for the graph-based approach (see chapter 4), and the determination of learning paths (see section 6) in form of the collaborative project *Lakeside IT-Curriculum*. In the first phase, the project aims to develop and test a curriculum for computer science and digital education for kindergarten and lower primary education. The development is based on the graph-based approach and the collected data presented in this dissertation. From the pool of the seven analyzed curricula (see section 3.5) the project team selected nine *target competencies* and five *support competencies*. In view of these competences, their dependencies are determined and represented in the form of tree-like structures. Three examples of these dependency structures are discussed in this chapter. The resulting curriculum is described and also some extracts are discussed.

During the project the participating educators from kindergarten filled out a small questionnaire to evaluate the curriculum. Following their opinion the curriculum is suitable and applicable for kindergarten, and covers the most important topics in the area of computer science and digital education. They think that the children are able to reach the included competencies. Only the order of the competencies should be revised.

A comparison of the dependency structures from this project and the determined learning paths from the *GECKO* platform showed that from overall 14 structures selected by experts at most 12 match with learning paths. The majority of exact matches are found using the method *covering central nodes* and the metric *PageRank*.

This project will be going on as the next phases, so primary and secondary education, are currently under development. The graph-based approach again lays out the foundation for finding *target competencies*. In further steps workshop units are planned and materials will be prepared.

Part IV

FUTURE WORK AND CONCLUSION

10. FUTURE WORK

10.1 Introduction

This work presents an approach to generate learning paths based on existing data from international curricula, educational models, and competency models. With that, it builds the basis for further development in larger projects. All described parts, like the analysis and comparison of curricula, the combination to a generic model, the online platform to organize and evaluate the data, and the tools to support the process, are designed to be extended in future work. The developments and extensions, that are possible and would make sense from the current perspective, are summarized in the following sections.

10.2 Educational Models

For this dissertation seven relevant educational models are selected to be part of the research. But there are a lot more models that are not considered because of different criteria. One example of such criteria are language limitations. With the help of new technologies in the field of automated translation of texts, or in this case competency formulations, more curricula, educational standards, and competency models can be added to the existing data set and be incorporated in the generic model. This will make the *Generic, Graph-Based Model for Competencies (GGBMC)* a dynamically growing environment which also includes recently published models from a large number of different countries.

Further, the data used in this dissertation is limited to the early years of education. Some of the selected models are developed for this age range and others are planned continuously, from kindergarten to graduation. In future development the data will be extended by adding competencies from higher levels. As a first extension the already included models with a continuous plan will be incorporated into the existing data and the *GGBMC*. Later development will also overcome the limits of secondary education and add competencies from undergraduate as well as post-graduate education.

As a further extension, curricula, educational standards, and competency models for other subjects like mathematics or languages can be added to the existing data. This would lead to a very large data set and the numbers of items in the *GGBMC*

would also increase. The great benefit of this would be the possibility to display dependencies of different subjects to each other. With this information interdisciplinary concepts can be fostered and it can be determined which prerequisites from other disciplines are necessary for given topic areas.

10.3 The Graph-based Approach

The graph-based approach aims at representing the educational models in a structured and comparable form. It is based on property graphs which allow attributes for the nodes as well as for the edges. Especially, the attributes enable future extensions by adding more information to the elements. One example for such an extension is the *prerequisitefactor* attribute which is already mentioned in section 4.4.2. With the possible values *hard* or *soft* this attribute gives information about the degree of the dependency. This means it is an attribute for relations which is already included in the graph-based model but not yet used. In future development, this factor can be used to give users even more information about prerequisites in learning paths. It might support the selection of the most suitable learning path.

Another extension could be weights for relations. These weights can be stored like attributes, but in form of numbers, to support calculations. Examples for such weights are *complexity factor*, an *estimated duration*, or an *importance value*. All these weights enable possibilities to determine learning paths for different purposes. Resulting learning paths can be the 'least complex' or the 'fastest' path. Adding all this information to the model would need a lot of time and work from experts as experience is crucial for appropriate estimations.

10.4 Software Development

10.4.1 GECKO

The *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*¹, to organize and evaluate the graph representation of the educational models, already offers many features. But, as it is mentioned in chapter 7, there are some elements missing or only partially implemented. One of the missing parts are the additional features for the role of a standard *user*. In this version of the platform *users* can register for free, take a look at the graphs, search and filter competencies in the graphs and determine learning paths without the possibility to store them. This is one of the main planned features for *users*. They will be able to collect desired competencies, store them in their profile and develop own learning paths for their use. The system will suggest competencies and point out dependencies

¹ <https://gecko.aau.at>

which will support the *users* in their development. In the navigation bar, which is specific for this role, the *users* will be able to manage their individual learning paths and curricula. The system is planned like a shopping cart where *users* chooses a competency from the graph and press a button 'add to cart'. When the *user* is finished he/she will be asked if the whole *prerequisites graph* or *learning paths* for the selected competencies should be added or not. This will be able for already individually created curricula.

Another role which can be added in future work is that of a *student*. Users with this role are part of a course which uses the *GECKO* platform and follows a determined learning path. They can get information about their reached level and which one they are targeting next.

In the latest version of the *GECKO* platform the three different layout options *manual layout*, *circle layout*, and *multicircle layout* are provided. They are very helpful in displaying information about the structure of the curricula. But there are several other possibilities to order the competencies to give information about specific situations. For example, a dependency-tree-like structure would be of interest showing roots and leaves of the graph. Further, the option of grouping competencies based on also selectable criteria like age groups or categories is of great interest. Here, a large number of extensions is possible to support the analysis.

A feature that can also be improved is the evaluation process for *experts*. Based on calculations from the support tools *experts* will get suggestions for relations between competencies if they rated a relation as incorrect. This feature would also deliver additional information for evaluation. As already mentioned in the future work for the graph-based approach, additional attributes of competencies and relations will be added. They have to be supported by the *GECKO* platform to enable a contribution or evaluation by the *experts*. As the evaluation takes a lot of time and is often a bit boring, an application for smartphones can be an option. Such an application can add elements like daily tasks and remind *experts* to evaluate some elements combined with some kind of reward system. The evaluation should have a very simple interface which allows to wipe to the right for 'OK' and to the left for 'Not OK'.

10.4.2 Supportive Tools

The tools to support steps like the generation of relations between competencies, the categorization of competencies, or the combination of competencies to *intersector nodes* offer opportunities for improvement or extension. All of them use basic measures and methods which possibly can be improved by more complex techniques. In the approach to identify relations between the competencies by similarity measures only *expands* relations are found. For the identification of *requires* relations extensions like an *ontology* are necessary. The development of this extension will

be part of future projects as well as the test and evaluation of the results. A way to add directions to the identified relations is also an important aspect for future development. As mentioned in section 8.5 the first approach using *Bloom's revised taxonomy* and defined action verbs (see section 2.5) does not deliver good results. Here, approaches, which also use the *ontology* or a different taxonomy, will be tested in future work. A further improvement in terms of relation identification is to consider transitive relations. These relations consist between competencies which are of course related in their content but not directly linked. In the experts results this can happen if there are competencies between the two related but not linked competencies. Such relations are identified by the algorithms used in the approach as they can be very similar. This means these cases cannot be treated as *false positives*. Here, an extension to consider transitive relations would improve the results of the algorithms.

In terms of the categorization of the competencies into the five categories for computer science education a possible improvement can be reached by using probabilistic models. These are well known in case of text classification and machine learning. As they are applied in a lot of fields, there exist very good models for the purpose of classification. The only drawback in the case of curriculum research can be the size of the data sets as the probabilistic models often need a lot of data to train the models. It would be of great interest to train models to categorize the competencies and test and evaluate their results in future projects.

The approach to combine competencies which are linguistically similar to *intersector nodes* need an improvement. For this purpose the results from the first experiment based on linguistic similarities will be investigated to find reasons for the limited number of matches with the experts opinions.

A major goal concerning the presented supportive tools is the incorporation into the *GECKO* platform. So, the whole process of generating the graphs out of the curriculum data will be supported by this platform. It is intended to just upload a list of competencies with all relevant attributes to the platform. With the help of the discussed approach, relations will be generated by the system. These relations have to be evaluated by experts. This is why the approach is called semi-automated. Additionally, the competencies are categorized into the five categories for computer science education by the system. As a last step also the *intersector nodes* are generated automatically. To reach this goal, some approaches have to be included in the implementation of the platform and others have to be improved to reach acceptable results before they can also be included.

10.5 Projects

As mentioned in chapter 9, the described and evaluated kindergarten project represents the first phase of the larger project called *Lakeside IT-Curriculum*. This project aims to develop, test, and implement a continuous curriculum for computer science and digital education starting in kindergarten until end of secondary education. For these aims, some preparatory work is necessary which is already mentioned in the future work for additional educational models. The generic model needs to be extended by competencies from higher school grades to determine suitable learning paths and transform them for the project's needs. This process requires a lot of discussion in the project team as well as with experts from the corresponding grades. Based on the results from the analysis of the curricula, the learning path generation, and the transformation for the project, activities in form of workshops have to be planned and documented. The material has also to be created, tested and revised in several pilot phases.

For the project in kindergarten the results from the survey of the educators' opinions have to be incorporated in the developed curriculum. Especially, opinions concerning individual competencies have to be considered in a revision. Also the mentioned sequences in the workshop activities have to be revised. These opinions of the educators also provide information for the development of the curriculum and the workshops for higher school grades.

10.6 Summary

This chapter summarizes future work for different parts of this dissertation. As it can be seen, the approaches and their evaluations presented build the basis for further development. The graph-based approach as well as the *Generic, Graph-Based Model for Competencies (GGBMC)* offer such opportunities for extensions. This can be additional attributes for competencies or relations or even the addition of whole school grades. Also, the developed software shows possibilities to be extended or improved. As a major goal all developed tools should work together on the *GECKO* online platform to simplify the incorporation of additional data. The project *Lakeside IT-Curriculum* is not yet finished as currently the early education is covered. As it is planned to be a continuous curriculum, in future development also higher school grades will be considered.

11. CONCLUSION

This dissertation follows the two objectives (a) to introduce a technique and framework for comprehensibly evaluating different curricula, standards and competency models, and (b) to prepare the ground for individually forming a computer science curriculum in primary and lower secondary schools. Based on these two objectives, a graph-based approach to formally analyze and compare curricula, educational standards and competency models is developed and presented. The results from this process and the comparison are discussed and evaluated. As a next step the educational models are combined to a *Generic, Graph-Based Model for Competencies (GGBMC)* to enable learning paths determination and optimization based on graph-theoretic methods. In addition, supportive tools are developed to simplify the process of representing curricula as graphs and to organize and evaluate these representations. As a use case of the graph-based approach the collaborative project *Lakeside IT-Curriculum* is presented. This chapter provides answers to the research questions formulated in section 1 and summarizes the results from the steps taken during the work for the dissertation. The subquestion **RQ1** to **RQ4** are answered before the main question **RQ0** is discussed.

RQ1: Which relevant competency models, educational standards, curricula and additional third-party projects or initiatives for computer science in primary and secondary education (in the Austrian context) exist?

As a first step to answer this first research question theory and background of the different educational approaches *competency-* and *standards-based education* are discussed in chapter 2. Both approaches are based on the more general concept of *outcome-based education* and the implementation and understanding of them differs in an international comparison. But, it is shown that the learning outcomes in both approaches can, under given circumstances, be compared to each other. With that the basis to compare and analyze different educational models is build and the selection process of educational models is not limited to one type of education. This conclusion is necessary to answer *RQ1*.

In chapter 3 the results from research in related literature and informal interviews with experts show that there exist a lot of models for computer science education in primary and secondary education. In international sources the national curricula from Australia and England are cited often, as well as the computer science stan-

dards from the *Computer Science Teacher Association (CSTA)* from 2011. A new version of the CSTA standards were published in 2017 and because of the impact of the version from 2011 on computer science education and its recency also the new version is seen to be relevant. For the Austrian context the Austrian model for digital competency called *digikomp* and especially approaches from German-speaking neighbor countries are of interest. These are the national curriculum 21 from Switzerland and the standards for informatics from the *German Informatics Society (GI)*. Among the seven selected models there is one competency model (Austria), one national curriculum based on competencies (Switzerland), four educational standards (Australia, Germany, two from USA), and one national curriculum with additional 'I can ...'-statements (England). As the detailed analysis of the models shows, the criteria for a comparison between the *competency-* and *standards-based* ones are fulfilled. For the curriculum from England (which is not outcome-based) an additional model including mentioned 'I can ...'-statements is used for the purpose of comparison as these statements also meet specific criteria.

RQ2: To what extend is it possible to (semi-)automatically generate graph representations of curricula?

Research question 2 is answered by the documentation and evaluation of the developed platform called *Graph-based Environment for Competency and Knowledge-Item Organization (GECKO)*¹ in chapter 7 and the supportive tools in chapter 8. The *GECKO* platform is developed for the purpose to support the organization and evaluation of the graph-representations from the educational models. It can be accessed online and is planned to be free for standard users. Besides the main functionality, it provides the possibilities to display the graph-representations, edit the graphs, calculate graph-based metrics for the analysis, and to determine learning paths for individual curricula. This platform is based on data representation and requires manual input in form of competency and relation lists to display the graphs from the curricula, educational standards and competency models. Based on the lists and the included attributes the system generates graphs, and enables search and filtering options. But, it also contributes to the semi-automatically generation of graph representations in terms of combining competencies to *intersector nodes*. The competency and relation lists are uploaded for single curricula. To combine them to the *Generic, Graph-Based Model for Competencies (GGBMC)* again lists of *intersector nodes* and the combined competencies are necessary. With these lists the system generates the *intersector nodes*, combines competencies and automatically organizes the transformation of the corresponding relations. A crucial part of the generation of the graph representations are the expert evaluations which cannot be automated. The *GECKO* platform provides the possibility for experts to evaluate

¹ <https://gecko.aau.at>

relations and competencies online with support from the system. Experts do not have to do large parts at once. They can evaluate some relations or competencies and the progress is automatically stored and displayed by the system. Additionally, experts get an overview of which elements they already evaluated in the structure in the curricula. The system also shows basic statistics concerning the evaluations of the experts.

In chapter 8, additional approaches to semi-automatically generated graph representations are presented and evaluated. These approaches consider the three processes identification of relations between competencies, categorization of competencies, and combination of competencies to *intersector nodes* which are part of the generation process and turned out to be very time consuming. All of three approaches are based on *linguistic features* and *natural language processing*.

The approach to identify relations between competencies is based on *similarity measures*. The assumption is that linguistic similar competencies are somehow related. In a first test the two different *similarity measures cosine similarity* and *Jaccard coefficient* are tested on different part of speech combinations. It turned out that the most matches with the relations created and evaluated by experts are reached with the *Jaccard coefficient* and a combination of *nouns* and *verbs*. Based on these results some improvements like synonym replacement and part of speech weighting are added to the described methods. In terms of the measures *precision* and *recall* with this approach, the values 0.26 for *precision* and 0.44 for *recall* are reached. So, 44% of the relations considered by experts are found by this method. But, this approach is limited to *expands* relations as none of the *requires* relations are identified. Also, the directions of the relations are not part of this identification.

In terms of categorization two methods are tested which both are based on keywords found in the categories. The first method focuses on the decision if a competency belongs to *computer science* or *digital literacy*. It extracts keywords from well known curricula for both categories to create dictionaries. With the help of these dictionaries the competencies are categorized. Results from this approach are compared to the opinions of experts and evaluated. It turned out that 70% of the competencies match the opinions of the experts. Considering only those competencies which show a strong agreement from the experts, 90% are categorized according to the experts' opinions. A second method to categorize the competencies into the five categories for computer science education *digital applications*, *digital infrastructure*, *programming and algorithms*, *data representation*, and *human factors and ethics* [DB15] is based on the fact that 65.8% of the nouns from the competency formulations occur uniquely in one category. Again, dictionaries of keywords for the categories are generated. But, in this approach already categorized competencies are used as *training set*. Comparing the results for a *test set* to existing data shows that 49% of the categories match with the experts' opinions.

The approach to automatically combine competencies to *intersector nodes* is also

based on *linguistic similarity* between the competencies from all curricula. It also uses the *Jaccard coefficient* and focuses on *nouns* and *verbs*. Out of 88 *intersector nodes* only 13 are partially identified by the means of this approach. These are only 14% of all *intersector nodes*.

The results from the different approaches give some insights into the possibilities of the (semi-)automated generation of graph representations for curricula and show to what extend the steps of this process can be automated.

RQ3: How and to what extend can curricula specifications be compared based on graph-theory and is it possible to identify central competencies?

This research question is answered by presenting the *graph-based approach* in chapter 4 and the results from analysis and comparison in chapter 5. The *graph-based approach* represents competencies as nodes and their dependencies as *expands* or *requires* relations. Both components have several attributes to store information which is crucial for the approach and build a *property graph* for competencies. These graphs are mapped to a *graph database* to enable search and calculations. After revisions based on the evaluation of experts, the representations of the curricula are combined to a *Generic, Graph-based Model for Competencies*. This process requires a standardization of competency formulations, the categorization of the standardized competencies, and the combination of very similar competencies to *intersector nodes*. As they represent the linking points of the curricula, the *intersector nodes* are a crucial part of the model.

In chapter 5, the methods used to analyze and compare the graph representations of the curricula are discussed and the results are presented. It shows that the original curricula are very similar in their numbers of competencies but after the standardization process they differ by at most 69 competencies (Australian curriculum with 131, and the model for the English curriculum with 62 competencies). This reflects the differences in the competency formulations in terms of used verbs and objects. The numbers of relations in the curricula differ in the original as well as in the standardized versions which is also shown by the calculation of the graphs' *density* values.

To identify *central competencies*, the three different centrality measures *degree centrality*, *betweenness centrality*, and *PageRank* are used. Identified *central competencies* show special characteristics for all used measures. Competencies with high value in *in-degree* are necessary for a lot other competencies and are often very basic. High values in *out-degree* have competencies which connect paths in the graphs. As competencies with a high *out-degree* connect, also shorted paths, competencies with a high value in *betweenness centrality* are linking points for larger topic strands. A high value in *PageRank* can be found in very basic competencies which important competencies depend on.

Special nodes like *sinks* which have no outgoing relations, *sources* which have no

incoming relations, or *isolated nodes* which have no relations at all, give information about the structure of curricula. *Sinks* represent good entry points to start with a topic or learning path, with nodes identified as *sources* topics can end. *Isolated nodes* often indicate important topics of computer science which have in the first levels no direct connection to other topics. Additionally, the number of *connected components* show the interconnection of the topics respectively the competency paths in a curriculum. The results of this analysis show that the GI standards and the CSTA standards from 2017 have the highest number of independent topics whereas in the Australian curriculum all topics are connected.

The foci of the curricula are determined by a general categorization into *computer science* and *digital literacy* as well as a more detailed categorization into *digital applications*, *digital infrastructure*, *programming and algorithms*, *data representation*, and *human factors and ethics*. Results from the more general categorization show that the Austrian competency model, the curriculum from Switzerland, and the CSTA standards from 2011 have a trend towards *digital literacy* whereas the competencies from the Australian curriculum and the CSTA standards from 2017 are evenly distributed between the two categories. Considering the results from the more detailed categorization the majority of the competencies from the Austrian competency model and the CSTA standards from 2011 fall into *human factors and ethics*. The category with the highest number of competencies is in the Australian curriculum, the model for the English curriculum, and the CSTA standards from 2017 *programming and algorithms*, in the curriculum from Switzerland *digital applications*, and in the GI standards *digital infrastructure*.

With the presentation of the *graph-based approach* the first part of research question 3, how curricula specifications can be compared based on graph-theory, is answered. The comparison in terms of *basic metrics*, *central competencies*, *structure*, and *focus* shows opportunities and limits of an approach based on graph-theory.

RQ4: What is an optimal combination of competencies and competency-clusters to cover central competencies?

To answer this research question the single curricula are combined to the *Generic, Graph-Based Model for Competencies (GGBMC)* and it is analyzed using the same basic metrics and methods to identify *central competencies*. Additionally, *intersector nodes* are seen as *central competencies* as they occur in at least two different curricula. It shows that a third of all competencies are combined to *intersector nodes*. The detailed results from the analysis are described in chapter 6. As additional constructs *learning paths* and *prerequisites graphs* are introduced and defined to represent combinations of competencies which can be used as guideline for teaching sequences. Where *learning paths* highlight paths from one target competency to an already reached competency, the *prerequisites graphs* considers all dependencies for one target competency to a set of already reached competencies. To determine

the optimal *learning paths* the three methods *shortest path*, *covering central nodes*, and *highest overall centrality* are applied. Each of them delivers a set of *learning paths* because similar values can appear. The method *shortest path* determines the path with the lowest number of included competencies. *Covering central nodes* and *highest overall centrality* are based on *centrality measures*. As *covering central nodes* selects the paths which contain the nodes with the highest centrality values, *highest overall centrality* sums up the centrality values of all contained nodes and selects the paths with the highest sums. That means, for the purpose of covering basic competencies, *PageRank* is selected. *Learning paths*, which cover competencies a lot of other competencies depend on, are found using *in-degree centrality*. To cover competencies which connect larger topic strands, *betweenness centrality* can be used. If competencies should be covered which connect small competency paths *out-degree centrality* is the choice.

A use case for the graph-based approach and the determination of learning paths is found in the collaborative project *Lakeside IT-Curriculum* which has the goal to develop a continuous computer science and digital education curriculum starting at kindergarten and lasting till the end of secondary education. In its first phase, the kindergarten and first years of primary education are covered. The required development for the project is based on the approach and the data presented in this dissertation as from the pool of the seven analyzed curricula nine *target competencies* and five *support competencies* are selected by the project team. For each of these selected competencies the dependency structure is manually determined. Based on these dependency structures the methods to optimize learning paths are evaluated. The highest number of matches is reached by the combination of the method *covering central nodes* and the metric *PageRank*.

Optimal combinations of competencies and competency-clusters to cover *central competencies* are in case of this dissertation determined by three different methods which are partially based on *centrality measures*. The definition of 'optimal' depends on the situation and the purpose.

This now helps in answering the main research question.

RQ0: How can relevant central competencies for computer science education in primary and secondary school be identified and covered?

The work shows that summarized relevant *central competencies* can be found with the help of a graph-based approach representing curricula, educational standards, and competency models as graphs. Using graph-theoretic metrics and *centrality measures*, *central competencies* can be identified supported by experts opinions and developed software tools based on *natural language processing*. Which competencies should be covered strongly depends on the situation and the goals of teachers of curriculum developers. Methods based on *centrality measures* can support the determination of optimal *learning paths*.

APPENDIX

A. COMPETENCY LISTS

A.1 AT: Austria digikomp4 [Dig13b]

AT-1.0: The students are able to cite important application areas of information technology from the environment, they live in.

AT-2.0: The students are able to name areas where computers cannot replace people.

AT-3.0: The students are able to think about their use of digital media and talk about it with their parents and teachers.

AT-3.1: The students are able to think about their use of digital media.

AT-3.2: The students are able to talk about their use of digital media with their parents and teachers.

AT-4.0: The students are able to distinguish between real and virtual worlds.

AT-5.0: The students are able to design their digital self on the web.

AT-6.0: The students know that they leave traces on the Internet and are identifiable. They behave accordingly.

AT-6.1: The students know that they leave traces on the Internet.

AT-6.2: The students know that they are identifiable on the Internet.

AT-6.3: The students behave accordingly because they know that they leave traces on the Internet and are identifiable.

AT-7.0: The students are familiar with the basic rules and duties in dealing with their own and other people's data.

AT-7.1: The students are familiar with the basic rules in dealing with their own data.

AT-7.2: The students are familiar with the basic duties in dealing with their own data.

AT-7.3: The students are familiar with the basic rules in dealing with their other people's data.

AT-7.4: The students are familiar with the basic duties in dealing with other people's data.

AT-8.0: The students respect the copyright (music, film, pictures, texts, software) and the right to the protection of personal data, in particular the right to one's own image.

AT-8.1: The students respect the copyright (music, film, pictures, texts, software).

AT-8.2: The students respect the right to the protection of personal data.

AT-8.3: The students respect particular the right to one's own image.

AT-9.0: The students are aware of the risks associated with the use of information technologies and know how they should behave in a given case.

AT-9.1: The students are aware of the risks associated with the use of information technologies.

AT-9.2: The students know how they should behave in a given case concerning the risks associated with the use of information technologies.

AT-10.0: The students are aware of possible dangers in dealing with people they only know from the Internet and can get help.

AT-10.1: The students are aware of possible dangers in dealing with people they only know from the Internet.

AT-10.2: The students are able to get help concerning the dangers of dealing with people they only know from the Internet.

AT-11.0: The students are aware that business is also transacted on the Internet and that there are risks involved.

AT-11.1: The students are aware that business is also transacted on the Internet.

AT-11.2: The students are aware that there are risks involved with business on the Internet.

AT-12.0: The students are aware that there are dangers such as malware, especially when they exchange data or use the Internet.

AT-12.1: The students are aware that there are dangers such as malware.

AT-12.2: The students are aware that there are dangers such as malware, especially when they exchange data.

AT-12.3: The students are aware that there are dangers such as malware, especially when using the Internet.

AT-13.0: The students are familiar with how to protect their computers and who to turn to if they are in need.

AT-13.1: The students are familiar with how to protect their computers.

AT-13.2: The students know who to turn to when they need to protect their computer.

AT-14.0: The students know there is data to which they are not allowed to gain access and abusive access is punishable in some cases.

AT-14.1: The students know there is data to which they are not allowed to gain access.

AT-14.2: The students know abusive access to data, which they are not allowed to gain access, is punishable in some cases.

AT-15.0: The students are able to name some professions in which computer systems are important.

AT-16.0: The students are familiar with the historical development of communication technology in broad outlines.

- AT-17.0:** The students are able to name digital devices of daily life and use them responsibly.
- AT-17.1:** The students are able to name digital devices of everyday life.
- AT-17.2:** The students are able to use digital devices of daily life responsibly.
- AT-18.0:** The students are able to name and use storage media.
- AT-18.1:** The students are able to name storage media.
- AT-18.2:** The students are able to use storage media.
- AT-19.0:** The students are able to use digital devices and the Internet to learn.
- AT-19.1:** The students are able to use digital devices to learn.
- AT-19.2:** The students are able to use the Internet to learn.
- AT-20.0:** The students are able to start and shut down a computer.
- AT-20.1:** The students are able to start a computer.
- AT-20.2:** The students are able to shut down a computer.
- AT-21.0:** The students are able to properly log in and out.
- AT-21.1:** The students are able to properly log in.
- AT-21.2:** The students are able to properly log out.
- AT-22.0:** The students are able to start programs and use them.
- AT-22.1:** The students are able to start programs.
- AT-22.2:** The students are able to use programs.
- AT-23.0:** The students are able to save, retrieve and open files in a folder system.
- AT-23.1:** The students are able to save files in a folder system.
- AT-23.2:** The students are able to retrieve files in a folder system.
- AT-23.3:** The students are able to open files in a folder system.
- AT-24.0:** The students are able to insert, move, copy and delete files.
- AT-24.1:** The students are able to insert files.
- AT-24.2:** The students are able to move files.
- AT-24.3:** The students are able to copy files.
- AT-24.4:** The students are able to delete files.
- AT-25.0:** The students are able to use suitable platforms.
- AT-26.0:** The students are able to use networks to search and display information.
- AT-26.1:** The students are able to use networks to search information.
- AT-26.2:** The students are able to use networks to display information.
- AT-27.0:** The students are able to use networks to communicate.
- AT-28.0:** The students are able to use networks to collaborate.
- AT-29.0:** The students know that digital devices can be used differently and they are able to use them in everyday life.
- AT-29.1:** The students know that digital devices can be used differently.
- AT-29.2:** The students are able to use digital devices in everyday life.
- AT-30.0:** The students are able to enter texts and format them.
- AT-30.1:** The students are able to enter texts.
- AT-30.2:** The students are able to format texts.

- AT-31.0:** The students are able to copy, paste, move and delete items.
- AT-31.1:** The students are able to copy items.
- AT-31.2:** The students are able to paste items.
- AT-31.3:** The students are able to move items.
- AT-31.4:** The students are able to delete items.
- AT-32.0:** The students are able to correct texts and, if necessary, use spelling aids.
- AT-32.1:** The students are able to correct texts.
- AT-32.2:** The students are able to use spelling aids if necessary.
- AT-33.0:** The students are able to design their works with pictures and graphics and present them using media.
- AT-33.1:** The students are able to design their works with pictures and graphics.
- AT-33.2:** The students are able to present their works by using media.
- AT-34.0:** The students are able to create and design digital drawings and images.
- AT-34.1:** The students are able to create digital drawings and images.
- AT-34.2:** The students are able to design digital drawings and images.
- AT-35.0:** The students are able to use digital audio and video files.
- AT-35.1:** The students are able to use digital audio files.
- AT-35.2:** The students are able to use digital video files.
- AT-36.0:** The students are able to understand the structure of a table.
- AT-37.0:** The students are able to create and design a table.
- AT-37.1:** The students are able to create a table.
- AT-37.2:** The students are able to design a table.
- AT-38.0:** The students are able to perform age-appropriate calculations.
- AT-39.0:** The students are able to create a diagram.
- AT-40.0:** The students are able to know search engines for children and are able to use them.
- AT-40.1:** The students know search engines for children.
- AT-40.2:** The students are able to use search engines for children.
- AT-41.0:** The students are able to use information from the Internet in their work.
- AT-42.0:** The students are able to write, send and receive digital messages.
- AT-42.1:** The students are able to write digital messages.
- AT-42.2:** The students are able to send digital messages.
- AT-42.3:** The students are able to receive digital messages.
- AT-43.0:** The students are able to pay attention to manners on the Internet.
- AT-44.0:** The students are able to use digital tools to collaborate.
- AT-45.0:** The students are able to encrypt and decrypt some information from everyday life.
- AT-45.1:** The students are able to encrypt some information from everyday life.
- AT-45.2:** The students are able to decrypt some information from everyday life.
- AT-46.0:** The students are able to collect, store and modify data.
- AT-46.1:** The students are able to collect data.

AT-46.2: The students are able to store data.

AT-46.3: The students are able to modify data.

AT-47.0: The students are able to understand and execute simple instructions.

AT-47.1: The students are able to understand simple instructions.

AT-47.2: The students are able to execute simple instructions.

AT-48.0: The students are able to create simple instructions.

AT-49.0: The students know that a computer program is created by sequencing instructions.

A.2 AU: Australian National Curriculum [Aus13]

AU-1.0: The students are able to identify how common digital systems (hardware and software) are used to meet specific purposes.

AU-2.0: The students are able to use digital systems to represent simple patterns in data in different ways.

AU-3.0: The students are able to design solutions to simple problems using a sequence of steps and decisions.

AU-3.1: The students are able to design solutions to simple problems using a sequence of steps.

AU-3.2: The students are able to design solutions to simple problems using decisions.

AU-4.0: The students are able to collect familiar data and display them to convey meaning.

AU-4.1: The students are able to collect familiar data.

AU-4.2: The students are able to display collected data to convey meaning.

AU-5.0: The students are able to create and organise ideas and information using information systems, and share information in safe online environments.

AU-5.1: The students are able to create ideas using information systems.

AU-5.2: The students are able to create information using information systems.

AU-5.3: The students are able to organise ideas using information systems.

AU-5.4: The students are able to organise information using information systems.

AU-5.5: The students are able to share ideas and information with known people in safe online environments.

AU-6.0: The students are able to describe the purpose of familiar products, services and environments and how they meet a range of present needs.

AU-6.1: The students are able to describe the purpose of familiar products, services and environments.

AU-6.2: The students are able to describe how familiar products, services and environments meet a range of present needs.

AU-7.0: The students are able to list the features of technologies that influence

design decisions and identify how digital systems are used.

AU-7.1: The students are able to list the features of technologies that influence design decisions.

AU-7.2: The students are able to identify how digital systems are used.

AU-8.0: The students are able to identify needs, opportunities or problems and describe them.

AU-8.1: The students are able to identify needs.

AU-8.2: The students are able to describe identified needs.

AU-8.3: The students are able to identify opportunities.

AU-8.4: The students are able to describe identified opportunities.

AU-8.5: The students are able to identify problems.

AU-8.6: The students are able to describe identified problems.

AU-9.0: The students are able to collect, sort and display familiar data from a range of sources and recognise patterns in data.

AU-9.1: The students are able to collect familiar data from a range of sources.

AU-9.2: The students are able to sort familiar data from a range of sources.

AU-9.3: The students are able to display familiar data from a range of sources.

AU-9.4: The students are able to recognise patterns in data.

AU-10.0: The students are able to record design ideas using techniques including labelled drawings, lists and sequenced instructions.

AU-11.0: The students are able to (with guidance) produce designed solutions for each of the prescribed technologies contexts.

AU-12.0: The students are able to evaluate their ideas, information and solutions on the basis of personal preferences and provided criteria including care for the environment.

AU-12.1: The students are able to evaluate their ideas on the basis of personal preferences.

AU-12.2: The students are able to evaluate their ideas on the basis of provided criteria including care for the environment.

AU-12.3: The students are able to evaluate information on the basis of personal preferences.

AU-12.4: The students are able to evaluate information on the basis of provided criteria including care for the environment.

AU-12.5: The students are able to evaluate solutions on the basis of personal preferences.

AU-12.6: The students are able to evaluate solutions on the basis of provided criteria including care for the environment.

AU-13.0: The students are able to safely create solutions and communicate ideas and information face-to-face and online.

AU-13.1: The students are able to safely create solutions.

AU-13.2: The students are able to safely communicate ideas and information face-

to-face.

AU-13.3: The students are able to safely communicate ideas and information on-line.

AU-14.0: The students are able to describe how a range of digital systems (hardware and software) and their peripheral devices can be used for different purposes.

AU-15.0: The students are able to explain how the same data sets can be represented in different ways.

AU-16.0: The students are able to define simple problems, design and implement digital solutions using algorithms that involve decision-making and user input.

AU-16.1: The students are able to define simple problems.

AU-16.2: The students are able to design digital solutions using algorithms that involve decision-making.

AU-16.3: The students are able to implement digital solutions using algorithms that involve decision-making.

AU-16.4: The students are able to design digital solutions using algorithms that involve user input.

AU-16.5: The students are able to implement digital solutions using algorithms that involve user input.

AU-17.0: The students are able to explain how the solutions meet their purposes.

AU-18.0: The students are able to collect and manipulate different data when creating information and digital solutions.

AU-18.1: The students are able to collect different data when creating information and digital solutions.

AU-18.2: The students are able to manipulate different data when creating information and digital solutions.

AU-19.0: The students are able to safely use and manage information systems for identified needs using agreed protocols and describe how information systems are used.

AU-19.1: The students are able to safely use information systems for identified needs using agreed protocols.

AU-19.2: The students are able to safely manage information systems for identified needs using agreed protocols.

AU-19.3: The students are able to describe how information systems are used.

AU-20.0: The students are able to describe how social, technical and sustainability factors influence the design of solutions to meet present and future needs.

AU-20.1: The students are able to describe how social factors influence the design of solutions to meet present needs.

AU-20.2: The students are able to describe how social factors influence the design of solutions to meet future needs.

AU-20.3: The students are able to describe how technical factors influence the design of solutions to meet present needs.

AU-20.4: The students are able to describe how technical factors influence the design of solutions to meet future needs.

AU-20.5: The students are able to describe how sustainability factors influence the design of solutions to meet present needs.

AU-20.6: The students are able to describe how sustainability factors influence the design of solutions to meet future needs.

AU-21.0: The students are able to describe features of technologies that influence design decisions and how a range of digital systems can be used.

AU-21.1: The students are able to describe features of technologies that influence design decisions.

AU-21.2: The students are able to describe how a range of digital systems can be used.

AU-22.0: The students are able to outline and define needs, opportunities or problems.

AU-22.1: The students are able to outline needs.

AU-22.2: The students are able to define needs.

AU-22.3: The students are able to outline opportunities.

AU-22.4: The students are able to define opportunities.

AU-22.5: The students are able to outline problems.

AU-22.6: The students are able to define problems.

AU-23.0: The students are able to collect, manipulate and interpret data from a range of sources to support decisions.

AU-23.1: The students are able to collect data from a range of sources to support decisions.

AU-23.2: The students are able to manipulate data from a range of sources to support decisions.

AU-23.3: The students are able to interpret data from a range of sources to support decisions.

AU-24.0: The students are able to generate and record design ideas for an audience using technical terms and graphical and non-graphical representation techniques including algorithms.

AU-24.1: The students are able to generate design ideas for an audience using technical terms.

AU-24.2: The students are able to generate design ideas for an audience using graphical representation techniques.

AU-24.3: The students are able to generate design ideas for an audience using non-graphical representation techniques including algorithms.

AU-24.4: The students are able to record design ideas for an audience using technical terms.

AU-24.5: The students are able to record design ideas for an audience using graphical representation techniques.

AU-24.6: The students are able to record design ideas for an audience using non-graphical representation techniques including algorithms.

AU-25.0: The students are able to plan a sequence of steps (algorithms) to create solutions, including visual programs.

AU-26.0: The students are able to plan and safely produce designed solutions for each of the prescribed technologies contexts.

AU-26.1: The students are able to plan solutions for each of the prescribed technologies contexts.

AU-26.2: The students are able to safely produce designed solutions for each of the prescribed technologies contexts.

AU-27.0: The students are able to use identified criteria for success, including sustainability considerations, to judge the suitability of their ideas, solutions and processes.

AU-28.0: The students are able to use agreed protocols when collaborating, and creating and communicating ideas, information and solutions face-to-face and on-line.

AU-28.1: The students are able to use agreed protocols when collaborating.

AU-28.2: The students are able to use agreed protocols when creating ideas, information and solutions face-to-face.

AU-28.3: The students are able to use agreed protocols when creating ideas, information and solutions online.

AU-28.4: The students are able to use agreed protocols when communicating ideas, information and solutions face-to-face.

AU-28.5: The students are able to use agreed protocols when communicating ideas, information and solutions online.

AU-29.0: The students are able to explain the fundamentals of digital system components (hardware, software and networks) and how digital systems are connected to form networks.

AU-29.1: The students are able to explain the fundamentals of digital system components (hardware, software and networks).

AU-29.2: The students are able to explain how digital systems are connected to form networks.

AU-30.0: The students are able to explain how digital systems use whole numbers as a basis for representing a variety of data types.

AU-31.0: The students are able to define problems in terms of data and functional requirements and design solutions by developing algorithms to address the problems.

AU-31.1: The students are able to define problems in terms of data.

AU-31.2: The students are able to define problems in terms of functional requirements.

AU-31.3: The students are able to design solutions by developing algorithms to address self-defined problems.

AU-32.0: The students are able to incorporate decision-making, repetition and user interface design into their designs and implement their digital solutions, including a visual program.

AU-32.1: The students are able to incorporate decision-making into their designs.

AU-32.2: The students are able to incorporate repetition into their designs.

AU-32.3: The students are able to incorporate user interface design into their designs.

AU-32.4: The students are able to implement their digital solutions, including a visual program.

AU-33.0: The students are able to explain how information systems and their solutions meet needs and consider sustainability.

AU-33.1: The students are able to explain how information systems and their solutions meet needs.

AU-33.2: The students are able to explain how information systems and their solutions consider sustainability.

AU-34.0: The students are able to manage the creation and communication of ideas and information in collaborative digital projects using validated data and agreed protocols.

AU-34.1: The students are able to manage the creation of ideas and information in collaborative digital projects using validated data and agreed protocols.

AU-34.2: The students are able to manage the communication of ideas and information in collaborative digital projects using validated data and agreed protocols.

AU-35.0: The students are able to explain how social, ethical, technical and sustainability considerations influence the design of solutions to meet a range of present and future needs.

AU-35.1: The students are able to explain how social considerations influence the design of solutions to meet a range of present .

AU-35.2: The students are able to explain how social considerations influence the design of solutions to meet a range of future needs.

AU-35.3: The students are able to explain how ethical considerations influence the design of solutions to meet a range of present needs.

AU-35.4: The students are able to explain how ethical considerations influence the design of solutions to meet a range of future needs.

AU-35.5: The students are able to explain how technical considerations influence the design of solutions to meet a range of present needs.

AU-35.6: The students are able to explain how technical considerations influence the design of solutions to meet a range of future needs.

AU-35.7: The students are able to explain how sustainability considerations influence the design of solutions to meet a range of present needs.

AU-35.8: The students are able to explain how sustainability considerations influence the design of solutions to meet a range of future needs.

AU-36.0: The students are able to explain how the features of technologies influence design decisions and how digital systems are connected to form networks.

AU-36.1: The students are able to explain how the features of technologies influence design decisions.

AU-37.0: The students are able to describe a range of needs, opportunities or problems and define them in terms of functional requirements.

AU-37.1: The students are able to describe a range of needs, opportunities or problems.

AU-37.2: The students are able to define a range of self-defined needs, opportunities or problems in terms of functional requirements.

AU-38.0: The students are able to collect and validate data from a range of sources to assist in making judgments.

AU-38.1: The students are able to collect data from a range of sources to assist in making judgments.

AU-38.2: The students are able to validate data from a range of sources to assist in making judgments.

AU-39.0: The students are able to generate and record design ideas for specified audiences using appropriate technical terms, and graphical and non-graphical representation techniques including algorithms.

AU-39.1: The students are able to generate design ideas for specified audiences using appropriate technical terms.

AU-39.2: The students are able to generate design ideas for specified audiences using graphical representation techniques.

AU-39.3: The students are able to generate design ideas for specified audiences using non-graphical representation techniques including algorithms.

AU-39.4: The students are able to record design ideas for specified audiences using appropriate technical terms.

AU-39.5: The students are able to record design ideas for specified audiences using graphical representation techniques.

AU-39.6: The students are able to record design ideas for specified audiences using non-graphical representation techniques including algorithms.

AU-40.0: The students are able to plan, design, test, modify and create digital solutions that meet intended purposes including user interfaces and a visual program.

AU-40.1: The students are able to plan digital solutions that meet intended purposes including user interfaces and a visual program.

AU-40.2: The students are able to design digital solutions that meet intended purposes including user interfaces and a visual program.

AU-40.3: The students are able to test digital solutions that meet intended purposes including user interfaces and a visual program.

AU-40.4: The students are able to modify digital solutions that meet intended purposes including user interfaces and a visual program.

AU-40.5: The students are able to create digital solutions that meet intended purposes including user interfaces and a visual program.

AU-41.0: The students are able to plan and document processes and resources and safely produce designed solutions for each of the prescribed technologies contexts.

AU-41.1: The students are able to plan processes.

AU-41.2: The students are able to document processes.

AU-41.3: The students are able to plan resources.

AU-41.4: The students are able to document resources.

AU-41.5: The students are able to safely produce designed solutions for each of the prescribed technologies contexts.

AU-42.0: The students are able to negotiate criteria for success, including sustainability considerations, and use these to judge the suitability of their ideas, solutions and processes.

AU-42.1: The students are able to negotiate criteria for success, including sustainability considerations.

AU-42.2: The students are able to use criteria for success, including sustainability considerations, to judge the suitability of their ideas, solutions and processes.

AU-43.0: The students are able to use ethical, social and technical protocols when collaborating, and creating and communicating ideas, information and solutions face-to-face and online.

AU-43.1: The students are able to use ethical, social and technical protocols when collaborating.

AU-43.2: The students are able to use ethical, social and technical protocols when creating ideas, information and solutions face-to-face.

AU-43.3: The students are able to use ethical, social and technical protocols when creating ideas, information and solutions online.

AU-43.4: The students are able to use ethical, social and technical protocols when communicating ideas, information and solutions face-to-face.

AU-43.5: The students are able to use ethical, social and technical protocols when communicating ideas, information and solutions online.

A.3 CH: Switzerland Curriculum 21 [Leh14]

CH-1.0: The students are able to exchange experiences in their close environment, media experiences as well as experiences in virtual living spaces and talk about their use of media (e.g. outdoor, playground, film, television, picture book, audio book, e-learning).

CH-1.1: The students are able to exchange experiences in their close environment.

CH-1.2: The students are able to exchange media experiences.

CH-1.3: The students are able to exchange experiences in virtual living spaces.

CH-1.4: The students are able to talk about their use of media.

CH-2.0: The students are able to understand simple contributions in different media languages and are able to talk about them (text, picture, everyday symbol, sound, film).

CH-2.1: The students are able to understand simple contributions in different media languages (text, picture, everyday symbol, sound, film).

CH-2.2: The students are able to talk about simple contributions in different media languages (text, picture, everyday symbol, sound, film).

CH-3.0: The students are able to recognise advertising and talk about the aims of the advertising messages.

CH-3.1: The students are able to recognise advertising.

CH-3.2: The students are able to talk about the aims of the advertising messages.

CH-4.0: The students are able to name the immediate emotions that media use can trigger (e.g. joy, anger, grief).

CH-5.0: The students are able to learn using predefined media and obtain information on a specific topic (e.g. book, magazine, educational game, game history, website).

CH-5.1: The students are able to learn using predefined media.

CH-5.2: The students are able to obtain information on a specific topic.

CH-6.0: The students are able to experiment playfully and creatively with media.

CH-6.1: The students are able to experiment playfully with media.

CH-6.2: The students are able to experiment creatively with media.

CH-7.0: The students are able to create and present simple picture, text and sound documents.

CH-7.1: The students are able to create simple picture documents.

CH-7.2: The students are able to create simple text documents.

CH-7.3: The students are able to create simple sound documents.

CH-7.4: The students are able to present simple picture documents.

CH-7.5: The students are able to present simple text documents.

CH-7.6: The students are able to present simple sound documents.

CH-8.0: The students are able to use the media to maintain existing contacts and exchange information (e.g. telephone, letter).

CH-9.0: The students are able to sort things according to their own chosen properties so that they can find an object with a certain property more quickly (e.g. colour, shape, size).

CH-10.0: The students are able to recognise and follow formal instructions (e.g. cooking and baking recipes, play and craft instructions, dance choreographies).

CH-10.1: The students are able to recognise formal instructions (e.g. cooking and baking recipes, play and craft instructions, dance choreographies).

CH-10.2: The students are able to follow formal instructions (e.g. cooking and baking recipes, play and craft instructions, dance choreographies).

CH-11.0: The students are able to switch devices on and off, start, operate and exit programs, and use simple functions.

CH-11.1: The students are able to switch devices on.

CH-11.2: The students are able to switch devices off.

CH-11.3: The students are able to start programs.

CH-11.4: The students are able to operate programs.

CH-11.5: The students are able to exit programs.

CH-11.6: The students are able to use simple functions of programs.

CH-12.0: The students are able to log in to a local network or learning environment with their own login.

CH-12.1: The students are able to log in to a local network with their own login.

CH-12.2: The students are able to log in to a learning environment with their own login.

CH-13.0: The students are able to file and retrieve documents independently.

CH-13.1: The students are able to file documents independently.

CH-13.2: The students are able to retrieve documents independently.

CH-14.0: The students are able to work with basic elements of the user interface (windows, menu, several opened programs).

CH-15.0: The students are able to name the advantages and disadvantages of direct experiences through media or virtually transmitted experiences and justify their personal use of media.

CH-15.1: The students are able to name the advantages of direct experiences through media.

CH-15.2: The students are able to name the disadvantages of direct experiences through media.

CH-15.3: The students are able to name the advantages of virtually transmitted experiences.

CH-15.4: The students are able to name the disadvantages of virtually transmitted experiences.

CH-15.5: The students are able to justify their personal use of media.

CH-16.0: The students are able to recognise and name consequences of medial and virtual actions (e.g. identity formation, relationship management, cyberbullying).

CH-16.1: The students are able to recognise consequences of medial actions.

CH-16.2: The students are able to recognise consequences of virtual actions.

CH-16.3: The students are able to name consequences of medial actions.

CH-16.4: The students are able to name consequences of virtual actions.

CH-17.0: The students are able to identify the basic functions of media (information, education, opinion-forming, entertainment, communication).

CH-18.0: The students know mixed forms and are able to give typical examples (infotainment, edutainment).

CH-18.1: The students know mixed forms (infotainment, edutainment).

CH-18.2: The students are able to give typical examples (infotainment, edutainment).

CH-19.0: The students are able to obtain and select information from various sources and assess its quality and usefulness.

CH-19.1: The students are able to obtain from various sources.

CH-19.2: The students are able to select information from various sources.

CH-19.3: The students are able to assess the quality of information from various sources.

CH-19.4: The students are able to assess the usefulness of information from various sources.

CH-20.0: The students are able to use media to create and present their work (e.g. class newspaper, class blog, radio play, video clip).

CH-20.1: The students are able to use media to create their work.

CH-20.2: The students are able to use media to present their work.

CH-21.0: The students are able to include the security rules for handling personal data in their media content (e.g. personal details, password, nickname).

CH-22.0: The students are able to reuse media content and integrate it into their own productions (e.g. lecture, blog/class blog), stating the source.

CH-22.1: The students are able to reuse media content.

CH-22.2: The students are able to integrate media content into their own productions, stating the source.

CH-23.0: The students are able to use media to present their thoughts and knowledge to an audience and/or publish them.

CH-23.1: The students are able to use media to present their thoughts to an audience.

CH-23.2: The students are able to use media to present their knowledge to an audience.

CH-23.3: The students are able to use media to publish their thoughts.

CH-23.4: The students are able to use media to publish their knowledge.

CH-24.0: The students are able to estimate the effects of their own media contributions and consider them accordingly during production.

CH-24.1: The students are able to estimate the effects of their own media contributions.

CH-24.2: The students are able to consider the effects of their own media contributions during production.

CH-25.0: The students are able to use the media to work together and exchange opinions, following the safety rules.

CH-25.1: The students are able to use the media to work together.

CH-25.2: The students are able to use the media to exchange opinions.

CH-25.3: The students are able to follow the safety rules when using media.

CH-26.0: The students are able to communicate through the media, following the

rules of safety and conduct.

CH-26.1: The students are able to communicate through the media.

CH-26.2: The students are able to follow the rules of safety when communicating through the media.

CH-26.3: The students are able to follow the rules of conduct when communicating through the media.

CH-27.0: The students are able to use different forms of data representation (e.g. symbols, tables, graphs).

CH-28.0: The students are able to encrypt data using codes they have developed themselves.

CH-29.0: The students know analogue and digital representations of data (text, number, image and sound) and are able to assign the corresponding file types.

CH-29.1: The students know with analogue representations of data.

CH-29.2: The students know with digital representations of data.

CH-29.3: The students are able to assign the corresponding file types to analogue and digital representations of data.

CH-30.0: The students know the names of the document types they use.

CH-31.0: The students are able to recognize and use tree and network structures (e.g. folder structure on the computer, family tree, mind map, website).

CH-31.1: The students are able to recognize tree structures.

CH-31.2: The students are able to recognize network structures.

CH-31.3: The students are able to use tree structures.

CH-31.4: The students are able to use network structures.

CH-32.0: The students are able to understand how error-detecting and error-correcting codes work.

CH-32.1: The students are able to understand how error-detecting codes work.

CH-32.2: The students are able to understand how error-correcting codes work.

CH-33.0: The students are able to find solutions for simple problems and check whether they are correct by trying them out (e.g. looking for a way, developing a game strategy). They are able to compare different solutions.

CH-33.1: The students are able to find solutions for simple problems by trying them out.

CH-33.2: The students are able to test solutions for simple problems for correctness by trying them out.

CH-33.3: The students are able to compare different solutions.

CH-34.0: The students are able to recognise, describe and structure processes with loops and branches from their environment (e.g. using flow diagrams).

CH-34.1: The students are able to recognise processes with branches in their environment.

CH-34.2: The students are able to describe processes with branches in their environment.

- CH-34.3:** The students are able to recognise processes with loops in their environment.
- CH-34.4:** The students are able to describe processes with loops in their environment.
- CH-34.5:** The students are able to structure processes with branches in their environment.
- CH-34.6:** The students are able to structure processes with loops in their environment.
- CH-35.0:** The students are able to read and manually perform simple operations with loops, conditions and parameters.
- CH-35.1:** The students are able to read simple operations with conditions.
- CH-35.2:** The students are able to manually perform simple operations with conditions.
- CH-35.3:** The students are able to read simple operations with loops.
- CH-35.4:** The students are able to manually perform simple operations with loops.
- CH-35.5:** The students are able to read simple operations with parameters.
- CH-35.6:** The students are able to manually perform simple operations with parameters.
- CH-36.0:** The students are able to understand that a computer only can execute predefined instructions and that a program is a sequence of such instructions.
- CH-36.1:** The students are able to understand that a computer only can execute predefined instructions.
- CH-36.2:** The students are able to understand that a program is a sequence of predefined instructions.
- CH-37.0:** The students are able to write and test programs with loops, conditional instructions and parameters.
- CH-37.1:** The students are able to write programs with loops.
- CH-37.2:** The students are able to test programs with loops.
- CH-37.3:** The students are able to write programs with conditional Instructions.
- CH-37.4:** The students are able to test programs with conditional Instructions.
- CH-37.5:** The students are able to write programs with Parameters.
- CH-37.6:** The students are able to test programs with Parameters.
- CH-38.0:** The students are able to distinguish between operating system and application software.
- CH-39.0:** The students know different types of memory (e.g. hard disks, flash memory, main memory), their advantages and disadvantages and understand units of data size.
- CH-39.1:** The students know different types of memory (e.g. hard disks, flash memory, main memory).
- CH-39.2:** The students know advantages of different types of memory.
- CH-39.3:** The students know disadvantages of different types of memory.

- CH-39.4:** The students are able to understand units of data size.
- CH-40.0:** The students are able to apply solution strategies (e.g. help function, research) to problems with devices and programs.
- CH-40.1:** The students are able to apply solution strategies to problems with devices.
- CH-40.2:** The students are able to apply solution strategies to problems with programs.
- CH-41.0:** The students are able to explain how data can be lost and know the most important measures to protect themselves against it.
- CH-41.1:** The students are able to explain how data can be lost.
- CH-41.2:** The students know the most important measures to protect themselves against data loss.
- CH-42.0:** The students are able to understand the basic functionality of search engines.
- CH-43.0:** The students are able to distinguish between local devices, local networks and the Internet as storage locations for private and public data.
- CH-43.1:** The students are able to distinguish between local devices, local networks and the Internet as storage locations for private data.
- CH-43.2:** The students are able to distinguish between local devices, local networks and the Internet as storage locations for public data.
- CH-44.0:** The students are able to understand the performance units of information processing systems and can assess their relevance for specific applications (e.g. storage capacity, image resolution, computing capacity, data transfer rate).
- CH-44.1:** The students are able to understand the performance units of information processing systems.
- CH-44.2:** The students are able to assess the relevance of performance units of information processing systems for specific applications.

A.4 DE: GI Standards for Informatics in Primary Education *[Ges19]*

- DE-1.0:** The students are able to explain that documents consist of data.
- DE-2.0:** The students are able to represent information using data.
- DE-3.0:** The students are able to interpret data in order to gain information.
- DE-4.0:** The students are able to state that agreements are necessary to encode and decode data.
- DE-4.1:** The students are able to state that agreements are necessary to encode data.
- DE-4.2:** The students are able to state that agreements are necessary to decode data.

DE-5.0: The students are able to encode data into a binary representation and interpret binary elements as data.

DE-5.1: The students are able to encode data into a binary representation.

DE-5.2: The students are able to interpret binary elements as data.

DE-6.0: The students are able to execute algorithms in their environment.

DE-7.0: The students are able to use basic algorithmic components.

DE-8.0: The students are able to describe algorithms in everyday language.

DE-9.0: The students are able to describe automata in their environment as self-acting machines.

DE-10.0: The students are able to describe the states of automata.

DE-11.0: The students are able to describe their interaction with automata.

DE-12.0: The students are able to explain that an automata changes its states in a rule-controlled manner.

DE-13.0: The students are able to name the components of computer systems using the technical language of computer science.

DE-14.0: The students are able to describe that computer systems are designed by humans.

DE-15.0: The students are able to interact purposefully with information technology systems.

DE-16.0: The students are able to name and describe strategies to prevent data loss.

DE-16.1: The students are able to name strategies to prevent data loss.

DE-16.2: The students are able to describe strategies to prevent data loss.

DE-17.0: The students are able to explain that computer science is omnipresent in their world.

DE-18.0: The students are able to name ways to protect data from unwanted access.

DE-19.0: The students are able to follow rules in dealing with data and computer systems.

DE-19.1: The students are able to follow rules in dealing with computer systems.

DE-19.2: The students are able to follow rules in dealing with data.

DE-20.0: The students are able to explain that data can be person-related.

DE-21.0: The students are able to design their own binary encoding for a small number of different elements.

DE-22.0: The students are able to present information in different forms of representation (text, image, audio, video).

DE-23.0: The students are able to use and develop agreements to encrypt and decrypt data.

DE-23.1: The students are able to use agreements to encrypt data.

DE-23.2: The students are able to use agreements to decrypt data.

DE-23.3: The students are able to develop agreements to encrypt data.

- DE-23.4:** The students are able to develop agreements to decrypt data.
- DE-24.0:** The students are able to use and develop agreements for the transmission of messages.
- DE-24.1:** The students are able to use agreements for the transmission of messages.
- DE-24.2:** The students are able to develop agreements for the transmission of messages.
- DE-25.0:** The students are able to design, implement and test algorithms with the basic algorithmic blocks instruction, sequence, repetition and branching.
- DE-25.1:** The students are able to design algorithms with the basic algorithmic block instruction.
- DE-25.2:** The students are able to design algorithms with the basic algorithmic block sequence.
- DE-25.3:** The students are able to design algorithms with the basic algorithmic block repetition.
- DE-25.4:** The students are able to design algorithms with the basic algorithmic block branching.
- DE-25.5:** The students are able to implement algorithms with the basic algorithmic block instruction.
- DE-25.6:** The students are able to implement algorithms with the basic algorithmic block sequence.
- DE-25.7:** The students are able to implement algorithms with the basic algorithmic block repetition.
- DE-25.8:** The students are able to implement algorithms with the basic algorithmic block branching.
- DE-25.9:** The students are able to test algorithms with the basic algorithmic block instruction.
- DE-25.10:** The students are able to test algorithms with the basic algorithmic block sequence.
- DE-25.11:** The students are able to test algorithms with the basic algorithmic block repetition.
- DE-25.12:** The students are able to test algorithms with the basic algorithmic block branching.
- DE-26.0:** The students are able to present algorithms in various formal forms of representation.
- DE-27.0:** The students are able to compare algorithms using technical language.
- DE-28.0:** The students are able to program a computer science system.
- DE-29.0:** The students are able to describe states and state transitions of automata.
- DE-29.1:** The students are able to describe states of automata.
- DE-29.2:** The students are able to describe state transitions of automata.
- DE-30.0:** The students are able to create models of automata to accept (linguistic) inputs and generate (linguistic) outputs.

DE-30.1: The students are able to create models of automata to accept (linguistic) inputs.

DE-30.2: The students are able to create models of automata to generate (linguistic) outputs.

DE-31.0: The students are able to control automata by programming.

DE-32.0: The students are able to explain the need for a formal language for interaction with computer systems.

DE-33.0: The students are able to provide basic, general descriptions of the function and working methods of computer systems (IPO model).

DE-33.1: The students are able to provide basic, general descriptions of the function of computer systems (IPO model).

DE-33.2: The students are able to provide basic, general descriptions of the working methods of computer systems (IPO model).

DE-34.0: The students are able to save and retrieve data.

DE-34.1: The students are able to save data.

DE-34.2: The students are able to retrieve data.

DE-35.0: The students are able to distinguish between local and global data storage.

DE-36.0: The students are able to use procedures to back up data.

DE-37.0: The students are able to list the basic components of the Internet and describe how data is transferred on the Internet using fixed arrangements (protocols).

DE-37.1: The students are able to list the basic components of the Internet.

DE-37.2: The students are able to describe how data is transferred on the Internet using fixed arrangements (protocols).

DE-38.0: The students are able to name and describe the use of digital tools in school and their free time.

DE-38.1: The students are able to name the use of digital tools in school and their free time.

DE-38.2: The students are able to describe the use of digital tools in school and their free time.

DE-39.0: The students are able to take steps to protect data from unwanted access.

DE-40.0: The students are able to apply simple procedures to ensure the integrity of data.

DE-41.0: The students are able to develop and evaluate agreements on the use of data and computer systems.

DE-41.1: The students are able to develop agreements on the use of computer systems.

DE-41.2: The students are able to evaluate agreements on the use of computer systems.

DE-41.3: The students are able to develop agreements on the use of data.

DE-41.4: The students are able to evaluate agreements on the use of data.

DE-42.0: The students are able to explain that personal data can be collected and processed with computer systems.

DE-42.1: The students are able to explain that personal data can be collected with computer systems.

DE-42.2: The students are able to explain that personal data can be processed with computer systems.

A.5 GB: Model for the National Curriculum England [Ber15]

GB-1.0: Understand what algorithms are, how they are implemented as programs on digital devices, and that programs execute by following precise and unambiguous instructions.

GB-1.1: The students are able to explain what an algorithm is.

GB-1.2: The students are able to explain how algorithms are implemented as programs on digital devices.

GB-1.3: The students are able to explain how programs execute by following precise and unambiguous instructions.

GB-2.0: Create and debug simple programs.

GB-2.1: The students are able to create simple programs.

GB-2.2: The students are able to debug simple programs.

GB-3.0: Use logical reasoning to predict the behaviour of simple programs.

GB-3.1: The students are able to use logical reasoning to predict what will happen when I run a program.

GB-3.2: The students are able to use logical reasoning to predict what will happen when I read through computer code.

GB-4.0: Use technology purposefully to create, organise, store, manipulate and retrieve digital content.

GB-4.1: The students are able to use technology purposefully to retrieve digital content.

GB-4.2: The students are able to use technology purposefully to store digital content.

GB-4.3: The students are able to use technology purposefully to organise digital content.

GB-4.4: The students are able to use technology purposefully to create digital content.

GB-4.5: The students are able to use technology purposefully to manipulate digital content.

GB-5.0: The students are able to discuss common uses of information technology beyond school.

GB-6.0: Use technology safely and respectfully, keeping personal information private; identify where to go for help and support when they have concerns about content or contact on the internet or other online technologies.

GB-6.1: The students are able to explain how to keep personal information private.

GB-6.2: The students are able to explain how to use technology safely.

GB-6.3: The students are able to explain how to use technology respectfully.

GB-6.4: The students are able to explain where to go for help and support when pupils have concerns about content.

GB-6.5: The students are able to explain where to go for help and support when pupils have concerns about contact.

GB-7.0: Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts.

GB-7.1: The students are able to design programs that accomplish specific goals.

GB-7.2: The students are able to write programs that accomplish specific goals.

GB-7.3: The students are able to debug programs that accomplish specific goals.

GB-7.4: The students are able to control physical systems.

GB-7.5: The students are able to simulate physical systems.

GB-7.6: The students are able to solve problems by decomposing them into smaller parts.

GB-8.0: Use sequence, selection, and repetition in programs; work with variables and various forms of input and output.

GB-8.1: The students are able to use sequence in programs.

GB-8.2: The students are able to use selection in programs.

GB-8.3: The students are able to use repetition in programs.

GB-8.4: The students are able to work with variables.

GB-8.5: The students are able to work with various forms of input.

GB-8.6: The students are able to work with various forms of output.

GB-9.0: Use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs.

GB-9.1: The students are able to use logical reasoning to explain how some algorithms work.

GB-9.2: The students are able to use logical reasoning to detect errors in algorithms.

GB-9.3: The students are able to use logical reasoning to correct errors in algorithms.

GB-9.4: The students are able to use logical reasoning to detect errors in programs.

GB-9.5: The students are able to use logical reasoning to correct errors in programs.

GB-10.0: Understand computer networks, including the internet; how they can

provide multiple services, such as the World Wide Web, and the opportunities they offer for communication and collaboration.

GB-10.1: The students are able to explain how computer networks work, including the internet.

GB-10.2: The students are able to explain how computer networks can provide multiple services such as the World Wide Web.

GB-10.3: The students are able to understand what opportunities computer networks offer for communication.

GB-10.4: The students are able to understand what opportunities computer networks offer for collaboration.

GB-11.0: Use search technologies effectively, appreciate how results are selected and ranked, and be discerning in evaluating digital content.

GB-11.1: The students are able to explain how search results are selected.

GB-11.2: The students are able to explain how search results are ranked.

GB-11.3: The students are able to explain how to be discerning in evaluating digital content.

GB-11.4: The students are able to explain how to use search technologies effectively.

GB-12.0: Select, use and combine a variety of software (including internet services) on a range of digital devices to design and create a range of programs, systems and content that accomplish given goals, including collecting, analysing, evaluating and presenting data and information.

GB-12.1: The students are able to design a range of programs that accomplish given goals.

GB-12.2: The students are able to create a range of programs that accomplish given goals.

GB-12.3: The students are able to design a range of systems that accomplish given goals.

GB-12.4: The students are able to create a range of systems that accomplish given goals.

GB-12.5: The students are able to design a range of content that accomplish given goals.

GB-12.6: The students are able to create a range of content that accomplish given goals.

GB-12.7: The students are able to collect data.

GB-12.8: The students are able to analyse data.

GB-12.9: The students are able to evaluate data.

GB-12.10: The students are able to present data.

GB-12.11: The students are able to collect information.

GB-12.12: The students are able to analyse information.

GB-12.13: The students are able to evaluate information.

GB-12.14: The students are able to present information.

GB-12.15: The students are able to select a variety of software (including internet services) on a range of digital devices.

GB-12.16: The students are able to use a variety of software (including internet services) on a range of digital devices.

GB-12.17: The students are able to combine a variety of software (including internet services) on a range of digital devices.

GB-13.0: Use technology safely, respectfully and responsibly; recognise acceptable/unacceptable behaviour; identify a range of ways to report concerns about content and contact.

GB-13.1: The students are able to explain how to use technology responsibly.

GB-13.2: The students are able to discuss the difference between acceptable and unacceptable behaviour online.

A.6 US1: CSTA K-12 Computer Science Standards 2011 [SCF⁺11]

US1-1.0: The students are able to gather information and communicate electronically with others with support from teachers, family members or student partners.

US1-1.1: The students are able to gather information with support from teachers.

US1-1.2: The students are able to gather information with support from family members.

US1-1.3: The students are able to gather information with support from student partners.

US1-1.4: The students are able to communicate electronically with others with support from teachers.

US1-1.5: The students are able to communicate electronically with others with support from family members.

US1-1.6: The students are able to communicate electronically with others with support student partners.

US1-2.0: The students are able to work cooperatively and collaboratively with peers, teachers and others using technology.

US1-2.1: The students are able to work cooperatively and collaboratively with teachers.

US1-2.2: The students are able to work cooperatively and collaboratively with peers.

US1-2.3: The students are able to work cooperatively and collaboratively others.

US1-2.4: The students are able to work cooperatively and collaboratively with teachers using technology.

US1-2.5: The students are able to work cooperatively and collaboratively with peers using technology.

US1-2.6: The students are able to work cooperatively and collaboratively with others using technology.

US1-3.0: The students are able to use technology resources (e.g. puzzles, logical thinking programs) to solve age-appropriate problems.

US1-4.0: The students are able to use writing tools, digital cameras, and drawing tools to illustrate thoughts, ideas, and stories in a step-by-step manner.

US1-4.1: The students are able to use writing tools to illustrate thoughts in a step-by-step manner.

US1-4.2: The students are able to use digital cameras to illustrate thoughts in a step-by-step manner.

US1-4.3: The students are able to use drawing tools to illustrate thoughts in a step-by-step manner.

US1-4.4: The students are able to use writing tools to illustrate ideas in a step-by-step manner.

US1-4.5: The students are able to use digital cameras to illustrate ideas in a step-by-step manner.

US1-4.6: The students are able to use drawing tools to illustrate ideas in a step-by-step manner.

US1-4.7: The students are able to use writing tools to illustrate stories in a step-by-step manner.

US1-4.8: The students are able to use digital cameras to illustrate stories in a step-by-step manner.

US1-4.9: The students are able to use drawing tools to illustrate stories in a step-by-step manner.

US1-5.0: The students are able to understand how to arrange (sort) information in useful order, such as sorting students by birth date, without using a computer.

US1-5.1: The students are able to understand how to arrange (sort) information.

US1-5.2: The students are able to understand useful order, such as sorting students by birth date.

US1-6.0: The students are able to recognize that software is created to control computer operations.

US1-7.0: The students are able to demonstrate how 0s and 1s can be used to represent information.

US1-8.0: The students are able to use technology resources to conduct age-appropriate research.

US1-9.0: The students are able to use developmentally appropriate multimedia resources (e.g., interactive books and educational software) to support learning across the curriculum.

US1-10.0: The students are able to create developmentally appropriate multimedia products with support from teachers, family members or student partners.

US1-10.1: The students are able to create developmentally appropriate multimedia

products with support from teachers.

US1-10.2: The students are able to create developmentally appropriate multimedia products with support from family members.

US1-10.3: The students are able to create developmentally appropriate multimedia products with support from student partners.

US1-11.0: The students are able to construct a set of statements to be acted out to accomplish a simple task (e.g., turtle instructions).

US1-12.0: The students are able to identify jobs that use computing and technology.

US1-12.1: The students are able to identify jobs that use technology.

US1-12.2: The students are able to identify jobs that use computing.

US1-13.0: The students are able to gather and organize information using concept-mapping tools.

US1-13.1: The students are able to gather information using concept-mapping tools.

US1-13.2: The students are able to organize information using concept-mapping tools.

US1-14.0: The students are able to use standard input and output devices to successfully operate computers and related technologies.

US1-14.1: The students are able to use standard input devices to successfully operate computers.

US1-14.2: The students are able to use standard output devices to successfully operate computers.

US1-14.3: The students are able to use standard input devices to successfully operate computer related technologies.

US1-14.4: The students are able to use standard output devices to successfully operate computer related technologies.

US1-15.0: The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems and software.

US1-15.1: The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of technology systems.

US1-15.2: The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of software.

US1-16.0: The students are able to identify positive and negative social and ethical behaviors for using technology.

US1-16.1: The students are able to identify positive social behaviors for using technology.

US1-16.2: The students are able to identify negative social behaviors for using technology.

US1-16.3: The students are able to identify positive ethical behaviors for using technology.

US1-16.4: The students are able to identify negative ethical behaviors for using technology.

US1-17.0: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual and collaborative writing, communication and publishing activities.

US1-17.1: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual writing activities.

US1-17.2: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual communication activities.

US1-17.3: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual publishing activities.

US1-17.4: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for collaborative writing activities.

US1-17.5: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for collaborative communication activities.

US1-17.6: The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for collaborative publishing activities.

US1-18.0: The students are able to use online resources (e.g., email, online discussions, collaborative web environments) to participate in collaborative problem-solving activities for the purpose of developing solutions or products.

US1-18.1: The students are able to use online resources (e.g., email, online discussions, collaborative web environments) to participate in collaborative problem-solving activities for the purpose of developing solutions.

US1-18.2: The students are able to use online resources (e.g., email, online discussions, collaborative web environments) to participate in collaborative problem-solving activities for the purpose of developing products.

US1-19.0: The students are able to identify ways that teamwork and collaboration can support problem solving and innovation.

US1-19.1: The students are able to identify ways that teamwork can support problem solving.

US1-19.2: The students are able to identify ways that teamwork can support innovation.

US1-19.3: The students are able to identify ways that collaboration can support problem solving.

US1-19.4: The students are able to identify ways that collaboration can support innovation.

US1-20.0: The students are able to understand and use the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of

sample instances, design, implementation and testing).

US1-20.1: The students are able to understand the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of sample instances, design, implementation and testing).

US1-20.2: The students are able to use the basic steps in algorithmic problem-solving (e.g., problem statement and exploration, examination of sample instances, design, implementation and testing).

US1-21.0: The students are able to develop a simple understanding of an algorithm (e.g., search, sequence of events, or sorting) using computer-free exercises.

US1-22.0: The students are able to demonstrate how a string of bits can be used to represent alphanumeric information.

US1-23.0: The students are able to describe how a simulation can be used to solve a problem.

US1-24.0: The students are able to make a list of sub-problems to consider while addressing a larger problem.

US1-25.0: The students are able to understand the connections between computer science and other fields.

US1-26.0: The students are able to use technology resources (e.g., calculators, data collection probes, mobile devices, videos, educational software and web tools) for problem-solving and self-directed learning.

US1-26.1: The students are able to use technology resources (e.g., calculators, data collection probes, mobile devices, videos, educational software and web tools) for problem-solving.

US1-26.2: The students are able to use technology resources (e.g., calculators, data collection probes, mobile devices, videos, educational software and web tools) for self-directed learning.

US1-27.0: The students are able to use general-purpose productivity tools and peripherals to support personal productivity, remediate skill deficits, and facilitate learning.

US1-27.1: The students are able to use general-purpose productivity tools and peripherals to support personal productivity.

US1-27.2: The students are able to use general-purpose productivity tools and peripherals to remediate skill deficits.

US1-27.3: The students are able to use general-purpose productivity tools and peripherals to facilitate learning.

US1-28.0: The students are able to gather and manipulate data using a variety of digital tools.

US1-28.1: The students are able to gather data using a variety of digital tools.

US1-28.2: The students are able to manipulate data using a variety of digital tools.

US1-29.0: The students are able to construct a program as a set of step-by-step instructions to be acted out (e.g., make a peanut butter and jelly sandwich activ-

ity).

US1-30.0: The students are able to implement problem solutions using a block based visual programming language.

US1-31.0: The students are able to use computing devices to access remote information, communicate with others in support of direct and independent learning and pursue personal interests.

US1-31.1: The students are able to use computing devices to access remote information.

US1-31.2: The students are able to use computing devices to communicate with others in support of direct learning.

US1-31.3: The students are able to use computing devices to communicate with others in support of independent learning.

US1-31.4: The students are able to use computing devices to pursue personal interests.

US1-32.0: The students are able to navigate between webpages using hyperlinks and conduct simple searches using search engines.

US1-32.1: The students are able to navigate between webpages using hyperlinks.

US1-32.2: The students are able to conduct simple searches using search engines.

US1-33.0: The students are able to identify a wide range of jobs that require knowledge or use of computing.

US1-33.1: The students are able to identify a wide range of jobs that require knowledge of computing.

US1-33.2: The students are able to identify a wide range of jobs that require use of computing.

US1-34.0: The students are able to demonstrate an appropriate level of proficiency with keyboards and other input and output devices.

US1-34.1: The students are able to demonstrate an appropriate level of proficiency with keyboards.

US1-34.2: The students are able to demonstrate an appropriate level of proficiency with input and output devices.

US1-35.0: The students are able to understand the pervasiveness of computers and computing in daily life (e.g., voicemail, downloading videos and audio files, microwave ovens, thermostats, wireless Internet, mobile computing devices, GPS systems).

US1-35.1: The students are able to understand the pervasiveness of computers in daily life.

US1-35.2: The students are able to understand the pervasiveness of computing in daily life.

US1-36.0: The students are able to apply strategies for identifying simple hardware and software problems that may occur during use.

US1-36.1: The students are able to apply strategies for identifying simple hardware

problems that may occur during use.

US1-36.2: The students are able to apply strategies for identifying simple software problems that may occur during use.

US1-37.0: The students are able to identify that information is coming to the computer from many sources over a network.

US1-38.0: The students are able to identify factors that distinguish humans from machines.

US1-39.0: The students are able to recognize that computers model intelligent behavior (as found in robotics, speech and language recognition, and computer animation).

US1-40.0: The students are able to discuss basic issues related to responsible use of technology and information, and the consequences of inappropriate use.

US1-40.1: The students are able to discuss basic issues related to responsible use of technology.

US1-40.2: The students are able to discuss basic issues related to responsible use of information.

US1-40.3: The students are able to discuss basic issues related to the consequences of inappropriate use of technology and information.

US1-41.0: The students are able to identify the impact of technology (e.g., social networking, cyber bullying, mobile computing and communication, web technologies, cyber security and virtualization) on personal life and society.

US1-41.1: The students are able to identify the impact of technology on personal life.

US1-41.2: The students are able to identify the impact of technology society.

US1-42.0: The students are able to evaluate the accuracy, relevance, appropriateness, comprehensiveness, and biases that occur in electronic information sources.

US1-42.1: The students are able to evaluate the accuracy that occur in electronic information sources.

US1-42.2: The students are able to evaluate the relevance that occur in electronic information sources.

US1-42.3: The students are able to evaluate the appropriateness that occur in electronic information sources.

US1-42.4: The students are able to evaluate the comprehensiveness that occur in electronic information sources.

US1-42.5: The students are able to evaluate biases that occur in electronic information sources.

US1-43.0: The students are able to understand ethical issues that relate to computers and networks (e.g., equity of access, security, privacy, copyright and intellectual property).

US1-43.1: The students are able to understand ethical issues that relate to computers.

US1-43.2: The students are able to understand ethical issues that relate to networks.

A.7 US2: CSTA K-12 Computer Science Standards 2017 [CST17]

US2-1.0: The students are able to select and operate appropriate software to perform a variety of tasks, and recognize that users have different needs and preferences for the technology they use.

US2-1.1: The students are able to operate appropriate software to perform a variety of tasks.

US2-1.2: The students are able to select appropriate software to perform a variety of tasks.

US2-1.3: The students are able to recognize that users have different needs for the technology they use.

US2-1.4: The students are able to recognize that users have different preferences for the technology they use.

US2-2.0: The students are able to use appropriate terminology in identifying and describing the function of common physical components of computing systems (hardware).

US2-2.1: The students are able to use appropriate terminology in identifying the function of common physical components of computing systems (hardware).

US2-2.2: The students are able to use appropriate terminology in describing the function of common physical components of computing systems (hardware).

US2-3.0: The students are able to describe basic hardware and software problems using accurate terminology.

US2-3.1: The students are able to describe basic hardware problems using accurate terminology.

US2-3.2: The students are able to describe basic software problems using accurate terminology.

US2-4.0: The students are able to explain what passwords are and why we use them, and use strong passwords to protect devices and information from unauthorized access.

US2-4.1: The students are able to explain what passwords are.

US2-4.2: The students are able to explain why we use passwords.

US2-4.3: The students are able to use strong passwords to protect devices from unauthorized access.

US2-4.4: The students are able to use strong passwords to protect information from unauthorized access.

US2-5.0: The students are able to store, copy, search, retrieve, modify, and delete information using a computing device and define the information stored as data.

US2-5.1: The students are able to store information using a computing device.

US2-5.2: The students are able to copy information using a computing device.

US2-5.3: The students are able to search information using a computing device.

US2-5.4: The students are able to retrieve information using a computing device.

US2-5.5: The students are able to modify information using a computing device.

US2-5.6: The students are able to delete information using a computing device.

US2-5.7: The students are able to define the information stored as data.

US2-6.0: The students are able to collect and present the same data in various visual formats.

US2-6.1: The students are able to collect the same data in various visual formats.

US2-6.2: The students are able to present the same data in various visual formats.

US2-7.0: The students are able to identify and describe patterns in data visualizations, such as charts or graphs, to make predictions.

US2-7.1: The students are able to identify patterns in data visualizations, such as charts or graphs, to make predictions.

US2-7.2: The students are able to describe patterns in data visualizations, such as charts or graphs, to make predictions.

US2-8.0: The students are able to model daily processes by creating and following algorithms (sets of step-by-step instructions) to complete tasks.

US2-8.1: The students are able to model daily processes by following algorithms (sets of step-by-step instructions) to complete tasks.

US2-8.2: The students are able to model daily processes by creating algorithms (sets of step-by-step instructions) to complete tasks.

US2-9.0: The students are able to model the way programs store and manipulate data by using numbers or other symbols to represent information.

US2-9.1: The students are able to model the way programs store data by using numbers.

US2-9.2: The students are able to model the way programs manipulate data by using numbers.

US2-9.3: The students are able to model the way programs store data by using symbols to represent information.

US2-9.4: The students are able to model the way programs manipulate data by using symbols to represent information.

US2-10.0: The students are able to develop programs with sequences and simple loops, to express ideas or address a problem.

US2-10.1: The students are able to develop programs with sequences, to express ideas.

US2-10.2: The students are able to develop programs with sequences, to address a problem.

US2-10.3: The students are able to develop programs with simple loops, to express ideas.

US2-10.4: The students are able to develop programs with simple loops, to address a problem.

US2-11.0: The students are able to decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

US2-12.0: The students are able to develop plans that describe a program's sequence of events, goals, and expected outcomes.

US2-12.1: The students are able to develop plans that describe a program's sequence of events.

US2-12.2: The students are able to develop plans that describe a program's goals.

US2-12.3: The students are able to develop plans that describe a program's expected outcomes.

US2-13.0: The students are able to give attribution when using the ideas and creations of others while developing programs.

US2-13.1: The students are able to give attribution when using the ideas of others while developing programs.

US2-13.2: The students are able to give attribution when using the creations of others while developing programs.

US2-14.0: The students are able to debug (identify and fix) errors in an algorithm or program that includes sequences and simple loops.

US2-14.1: The students are able to identify errors in an algorithm that includes sequences.

US2-14.2: The students are able to fix errors in an algorithm that includes sequences.

US2-14.3: The students are able to identify errors in an algorithm that includes simple loops.

US2-14.4: The students are able to fix errors in an algorithm that includes simple loops.

US2-14.5: The students are able to identify errors in a program that includes sequences.

US2-14.6: The students are able to fix errors in a program that includes sequences.

US2-14.7: The students are able to identify errors in a program that includes simple loops.

US2-14.8: The students are able to fix errors in a program that includes simple loops.

US2-15.0: The students are able to using correct terminology, describe steps taken and choices made during the iterative process of program development.

US2-15.1: The students are able to using correct terminology, describe steps taken during the iterative process of program development.

US2-15.2: The students are able to using correct terminology, describe choices made during the iterative process of program development.

US2-16.0: The students are able to compare how people live and work before and

after the implementation or adoption of new computing technology.

US2-16.1: The students are able to compare how people live before and after the implementation or adoption of new computing technology.

US2-16.2: The students are able to compare how people work before and after the implementation or adoption of new computing technology.

US2-17.0: The students are able to work respectfully and responsibly with others online.

US2-17.1: The students are able to work respectfully with others online.

US2-17.2: The students are able to work responsibly with others online.

US2-18.0: The students are able to keep login information private, and log off of devices appropriately.

US2-18.1: The students are able to keep login information private.

US2-18.2: The students are able to log off of devices appropriately.

US2-19.0: The students are able to describe how internal and external parts of computing devices function to form a system.

US2-20.0: The students are able to model how computer hardware and software work together as a system to accomplish tasks.

US2-21.0: The students are able to determine potential solutions to solve simple hardware and software problems using common troubleshooting strategies.

US2-21.1: The students are able to determine potential solutions to solve simple hardware problems using common troubleshooting strategies.

US2-21.2: The students are able to determine potential solutions to solve simple software problems using common troubleshooting strategies.

US2-22.0: The students are able to model how information is broken down into smaller pieces, transmitted as packets through multiple devices over networks and the Internet, and reassembled at the destination.

US2-22.1: The students are able to model how information is broken down into smaller pieces.

US2-22.2: The students are able to model how information is transmitted as packets through multiple devices over networks.

US2-22.3: The students are able to model how information is transmitted as packets through multiple devices over the Internet.

US2-22.4: The students are able to model how information is reassembled at the destination.

US2-23.0: The students are able to discuss real-world cybersecurity problems and how personal information can be protected.

US2-23.1: The students are able to discuss real-world cybersecurity problems.

US2-23.2: The students are able to discuss how personal information can be protected.

US2-24.0: The students are able to organize and present collected data visually to highlight relationships and support a claim.

US2-24.1: The students are able to organize collected data visually to highlight relationships.

US2-24.2: The students are able to present collected data visually to highlight relationships.

US2-24.3: The students are able to organize collected data visually to support a claim.

US2-24.4: The students are able to present collected data visually to support a claim.

US2-25.0: The students are able to use data to highlight or propose cause-and-effect relationships, predict outcomes, or communicate an idea.

US2-25.1: The students are able to use data to highlight cause-and-effect relationships.

US2-25.2: The students are able to use data to propose cause-and-effect relationships.

US2-25.3: The students are able to use data to predict outcomes.

US2-25.4: The students are able to use data to communicate an idea.

US2-26.0: The students are able to compare and refine multiple algorithms for the same task and determine which is the most appropriate.

US2-26.1: The students are able to compare multiple algorithms for the same task.

US2-26.2: The students are able to refine multiple algorithms for the same task.

US2-26.3: The students are able to determine which algorithm for the same task is the most appropriate.

US2-27.0: The students are able to create programs that use variables to store and modify data.

US2-27.1: The students are able to create programs that use variables to store data.

US2-27.2: The students are able to create programs that use variables to modify data.

US2-28.0: The students are able to create programs that include sequences, events, loops, and conditionals.

US2-28.1: The students are able to create programs that include sequences.

US2-28.2: The students are able to create programs that include events.

US2-28.3: The students are able to create programs that include loops.

US2-28.4: The students are able to create programs that include conditionals.

US2-29.0: The students are able to decompose (break down) problems into smaller, manageable subproblems to facilitate the program development process.

US2-30.0: The students are able to modify, remix, or incorporate portions of an existing program into one's own work, to develop something new or add more advanced features.

US2-30.1: The students are able to modify portions of an existing program, to develop something new.

US2-30.2: The students are able to remix portions of an existing program, to develop something new.

US2-30.3: The students are able to incorporate portions of an existing program into one's own work, to develop something new.

US2-30.4: The students are able to modify portions of an existing program, to add more advanced features.

US2-30.5: The students are able to remix portions of an existing program, to add more advanced features.

US2-30.6: The students are able to incorporate portions of an existing program into one's own work, to add more advanced features.

US2-31.0: The students are able to use an iterative process to plan the development of a program by including others' perspectives and considering user preferences.

US2-31.1: The students are able to use an iterative process to plan the development of a program by including others' perspectives.

US2-31.2: The students are able to use an iterative process to plan the development of a program by considering user preferences.

US2-32.0: The students are able to observe intellectual property rights and give appropriate attribution when creating or remixing programs.

US2-32.1: The students are able to observe intellectual property rights.

US2-32.2: The students are able to give appropriate attribution when creating programs.

US2-32.3: The students are able to give appropriate attribution when remixing programs.

US2-33.0: The students are able to test and debug (identify and fix errors) a program or algorithm to ensure it runs as intended.

US2-33.1: The students are able to test an algorithm to ensure it runs as intended.

US2-33.2: The students are able to debug (identify and fix errors) an algorithm to ensure it runs as intended.

US2-33.3: The students are able to test a program to ensure it runs as intended.

US2-33.4: The students are able to debug (identify and fix errors) a program to ensure it runs as intended.

US2-34.0: The students are able to take on varying roles, with teacher guidance, when collaborating with peers during the design, implementation, and review stages of program development.

US2-34.1: The students are able to take on varying roles, with teacher guidance, when collaborating with peers during the design stage of program development.

US2-34.2: The students are able to take on varying roles, with teacher guidance, when collaborating with peers during the implementation stage of program development.

US2-34.3: The students are able to take on varying roles, with teacher guidance, when collaborating with peers during the review stage of program development.

US2-35.0: The students are able to describe choices made during program development using code comments, presentations, and demonstrations

US2-35.1: The students are able to describe choices made during program development using code comments.

US2-35.2: The students are able to describe choices made during program development using presentations.

US2-35.3: The students are able to describe choices made during program development using demonstrations.

US2-36.0: The students are able to discuss computing technologies that have changed the world, and express how those technologies influence, and are influenced by, cultural practices.

US2-36.1: The students are able to discuss computing technologies that have changed the world.

US2-36.2: The students are able to express how computing technologies influence cultural practices.

US2-36.3: The students are able to express how computing technologies are influenced by cultural practices.

US2-37.0: The students are able to brainstorm ways to improve the accessibility and usability of technology products for the diverse needs and wants of users.

US2-37.1: The students are able to brainstorm ways to improve the accessibility of technology products for the diverse needs and wants of users.

US2-37.2: The students are able to brainstorm ways to improve the usability of technology products for the diverse needs and wants of users.

US2-38.0: The students are able to seek diverse perspectives for the purpose of improving computational artifacts.

US2-39.0: The students are able to use public domain or creative commons media, and refrain from copying or using material created by others without permission.

US2-39.1: The students are able to use public domain or creative commons media.

US2-39.2: The students are able to refrain from copying or using material created by others without permission.

B. INTERSECTOR NODES LISTS

B.1 Data Representation

INT-001-DR:

1. **AU-4.2:** The students are able to display collected data to convey meaning.
2. **CH-27.0:** The students are able to use different forms of data representation (e.g. symbols, tables, graphs).
3. **DE-22.0:** The students are able to present information in different forms of representation (text, image, audio, video).

INT-002-DR:

1. **GB-12.10:** The students are able to present data.
2. **US2-6.2:** The students are able to present the same data in various visual formats.

INT-003-DR:

1. **AU-9.4:** The students are able to recognise patterns in data.
2. **US2-7.1:** The students are able to identify patterns in data visualizations, such as charts or graphs, to make predictions.

INT-004-DR:

1. **DE-5.1:** The students are able to encode data into a binary representation.
2. **US1-7.0:** The students are able to demonstrate how 0s and 1s can be used to represent information.

INT-005-DR:

1. **AU-30.0:** The students are able to explain how digital systems use whole numbers as a basis for representing a variety of data types.

2. **US1-22.0:** The students are able to demonstrate how a string of bits can be used to represent alphanumeric information.

INT-006-DR:

1. **CH-28.0:** The students are able to encrypt data using codes they have developed themselves.
2. **DE-23.3:** The students are able to develop agreements to encrypt data.

B.2 Digital Applications**INT-001-DA:**

1. **AU-11.0:** The students are able to (with guidance) produce designed solutions for each of the prescribed technologies contexts.
2. **US1-10.1:** The students are able to create developmentally appropriate multimedia products with support from teachers.
3. **US1-10.2:** The students are able to create developmentally appropriate multimedia products with support from family members.
4. **US1-10.3:** The students are able to create developmentally appropriate multimedia products with support from student partners.

INT-002-DA:

1. **AT-30.1:** The students are able to enter texts.
2. **AT-30.2:** The students are able to format texts.
3. **US1-17.1:** The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual writing activities.

INT-003-DA:

1. **AU-5.1:** The students are able to create ideas using information systems.
2. **AU-5.2:** The students are able to create information using information systems.
3. **CH-7.1:** The students are able to create simple picture documents.
4. **CH-7.2:** The students are able to create simple text documents.

5. **CH-7.3:** The students are able to create simple sound documents.
6. **GB-4.4:** The students are able to use technology purposefully to create digital content.

INT-004-DA:

1. **AT-33.2:** The students are able to present their works by using media.
2. **CH-20.2:** The students are able to use media to present their work.
3. **US1-17.3:** The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for individual publishing activities.

INT-005-DA:

1. **AT-8.1:** The students respect the copyright (music, film, pictures, texts, software).
2. **AT-8.3:** The students respect particular the right to one's own image.
3. **CH-22.1:** The students are able to reuse media content.
4. **US2-39.1:** The students are able to use public domain or creative commons media.

INT-006-DA:

1. **CH-22.2:** The students are able to integrate media content into their own productions, stating the source.
2. **US2-39.2:** The students are able to refrain from copying or using material created by others without permission.

INT-007-DA:

1. **AT-29.1:** The students know that digital devices can be used differently.
2. **GB-12.15:** The students are able to select a variety of software (including internet services) on a range of digital devices.
3. **US2-1.2:** The students are able to select appropriate software to perform a variety of tasks.

INT-008-DA:

1. **AT-29.2:** The students are able to use digital devices in everyday life.
2. **GB-12.16:** The students are able to use a variety of software (including internet services) on a range of digital devices.
3. **US2-1.1:** The students are able to operate appropriate software to perform a variety of tasks.

INT-009-DA:

1. **AU-4.1:** The students are able to collect familiar data.
2. **CH-5.2:** The students are able to obtain information on a specific topic.

INT-010-DA:

1. **CH-42.0:** The students are able to understand the basic functionality of search engines.
2. **GB-11.1:** The students are able to explain how search results are selected.
3. **GB-11.2:** The students are able to explain how search results are ranked.

INT-011-DA:

1. **AT-26.1:** The students are able to use networks to search information.
2. **AU-9.1:** The students are able to collect familiar data from a range of sources.
3. **GB-4.1:** The students are able to use technology purposefully to retrieve digital content.
4. **US2-5.3:** The students are able to search information using a computing device.

INT-012-DA:

1. **AT-40.2:** The students are able to use search engines for children.
2. **US1-8.0:** The students are able to use technology resources to conduct age-appropriate research.

INT-013-DA:

1. **AT-19.1:** The students are able to use digital devices to learn.
2. **AT-19.2:** The students are able to use the Internet to learn.

3. **US1-9.0:** The students are able to use developmentally appropriate multi-media resources (e.g., interactive books and educational software) to support learning across the curriculum.

INT-014-DA:

1. **AT-46.1:** The students are able to collect data.
2. **AU-18.1:** The students are able to collect different data when creating information and digital solutions.
3. **DE-34.1:** The students are able to save data.
4. **GB-12.7:** The students are able to collect data.
5. **US1-28.1:** The students are able to gather data using a variety of digital tools.

INT-015-DA:

1. **AU-38.1:** The students are able to collect data from a range of sources to assist in making judgments.
2. **CH-19.1:** The students are able to obtain from various sources.

INT-016-DA:

1. **AU-38.2:** The students are able to validate data from a range of sources to assist in making judgments.
2. **CH-19.3:** The students are able to assess the quality of information from various sources.
3. **US1-42.1:** The students are able to evaluate the accuracy that occur in electronic information sources.

INT-017-DA:

1. **CH-19.4:** The students are able to assess the usefulness of information from various sources.
2. **US1-42.2:** The students are able to evaluate the relevance that occur in electronic information sources.

INT-018-DA:

1. **CH-9.0:** The students are able to sort things according to their own chosen properties so that they can find an object with a certain property more quickly (e.g. colour, shape, size).
2. **US1-5.1:** The students are able to understand how to arrange (sort) information.
3. **US1-5.2:** The students are able to understand useful order, such as sorting students by birth date.

INT-019-DA:

1. **AT-46.2:** The students are able to store data.
2. **GB-4.2:** The students are able to use technology purposefully to store digital content.
3. **US2-5.1:** The students are able to store information using a computing device.

INT-020-DA:

1. **AU-5.3:** The students are able to organise ideas using information systems.
2. **AU-5.4:** The students are able to organise information using information systems.
3. **DE-34.2:** The students are able to retrieve data.
4. **GB-4.3:** The students are able to use technology purposefully to organise digital content.
5. **US2-5.2:** The students are able to copy information using a computing device.
6. **US2-5.4:** The students are able to retrieve information using a computing device.
7. **US2-5.6:** The students are able to delete information using a computing device.

INT-021-DA:

1. **AT-46.3:** The students are able to modify data.
2. **GB-4.5:** The students are able to use technology purposefully to manipulate digital content.

3. **US2-5.5:** The students are able to modify information using a computing device.

INT-022-DA:

1. **AU-18.2:** The students are able to manipulate different data when creating information and digital solutions.
2. **US1-28.2:** The students are able to manipulate data using a variety of digital tools.

*B.3 Digital Infrastructure***INT-001-DI:**

1. **AT-20.1:** The students are able to start a computer.
2. **CH-11.1:** The students are able to switch devices on.

INT-002-DI:

1. **AT-20.2:** The students are able to shut down a computer.
2. **CH-11.2:** The students are able to switch devices off.

INT-003-DI:

1. **AT-22.1:** The students are able to start programs.
2. **CH-11.3:** The students are able to start programs.

INT-004-DI:

1. **AT-22.2:** The students are able to use programs.
2. **CH-11.4:** The students are able to operate programs.

INT-005-DI:

1. **CH-14.0:** The students are able to work with basic elements of the user interface (windows, menu, several opened programs).
2. **DE-15.0:** The students are able to interact purposefully with information technology systems.

INT-006-DI:

1. **AT-21.1:** The students are able to properly log in.
2. **CH-12.2:** The students are able to log in to a learning environment with their own login.

INT-007-DI:

1. **AT-23.1:** The students are able to save files in a folder system.
2. **CH-13.1:** The students are able to file documents independently.

INT-008-DI:

1. **AT-23.2:** The students are able to retrieve files in a folder system.
2. **CH-13.2:** The students are able to retrieve documents independently.

INT-009-DI:

1. **DE-13.0:** The students are able to name the components of computer systems using the technical language of computer science.
2. **US2-2.2:** The students are able to use appropriate terminology in describing the function of common physical components of computing systems (hardware).

INT-010-DI:

1. **AU-14.0:** The students are able to describe how a range of digital systems (hardware and software) and their peripheral devices can be used for different purposes.
2. **US2-19.0:** The students are able to describe how internal and external parts of computing devices function to form a system.

INT-011-DI:

1. **AU-29.1:** The students are able to explain the fundamentals of digital system components (hardware, software and networks).
2. **DE-33.2:** The students are able to provide basic, general descriptions of the working methods of computer systems (IPO model).
3. **US2-20.0:** The students are able to model how computer hardware and software work together as a system to accomplish tasks.

INT-012-DI:

1. **AT-18.1:** The students are able to name storage media.
2. **CH-39.1:** The students know different types of memory (e.g. hard disks, flash memory, main memory).

INT-013-DI:

1. **CH-40.1:** The students are able to apply solution strategies to problems with devices.
2. **US2-21.1:** The students are able to determine potential solutions to solve simple hardware problems using common troubleshooting strategies.

INT-014-DI:

1. **CH-40.2:** The students are able to apply solution strategies to problems with programs.
2. **US2-21.2:** The students are able to determine potential solutions to solve simple software problems using common troubleshooting strategies.

INT-015-DI:

1. **CH-41.1:** The students are able to explain how data can be lost.
2. **DE-16.1:** The students are able to name strategies to prevent data loss.

INT-016-DI:

1. **CH-41.2:** The students know the most important measures to protect themselves against data loss.
2. **DE-16.2:** The students are able to describe strategies to prevent data loss.

INT-017-DI:

1. **AU-29.2:** The students are able to explain how digital systems are connected to form networks.
2. **DE-37.1:** The students are able to list the basic components of the Internet.
3. **GB-10.1:** The students are able to explain how computer networks work, including the internet.

INT-018-DI:

1. **DE-37.2:** The students are able to describe how data is transferred on the Internet using fixed arrangements (protocols).

2. **GB-10.2:** The students are able to explain how computer networks can provide multiple services such as the World Wide Web.

INT-019-DI:

1. **CH-43.1:** The students are able to distinguish between local devices, local networks and the Internet as storage locations for private data.
2. **DE-35.0:** The students are able to distinguish between local and global data storage.

B.4 Human Factors and Ethics**INT-001-HF:**

1. **AT-17.1:** The students are able to name digital devices of everyday life.
2. **DE-38.1:** The students are able to name the use of digital tools in school and their free time.

INT-002-HF:

1. **AT-3.2:** The students are able to talk about their use of digital media with their parents and teachers.
2. **CH-1.4:** The students are able to talk about their use of media.

INT-003-HF:

1. **US1-41.1:** The students are able to identify the impact of technology on personal life.
2. **US2-16.1:** The students are able to compare how people live before and after the implementation or adoption of new computing technology.

INT-004-HF:

1. **US1-41.2:** The students are able to identify the impact of technology society.
2. **US2-16.2:** The students are able to compare how people work before and after the implementation or adoption of new computing technology.

INT-005-HF:

1. **AT-43.0:** The students are able to pay attention to manners on the Internet.

2. **AU-19.1:** The students are able to safely use information systems for identified needs using agreed protocols.
3. **DE-19.1:** The students are able to follow rules in dealing with computer systems.

INT-006-HF:

1. **AT-17.2:** The students are able to use digital devices of daily life responsibly.
2. **US1-15.2:** The students are able to practice responsible digital citizenship (legal and ethical behaviors) in the use of software.

INT-007-HF:

1. **GB-13.2:** The students are able to discuss the difference between acceptable and unacceptable behaviour online.
2. **US1-40.2:** The students are able to discuss basic issues related to responsible use of information.

INT-008-HF:

1. **AT-16.0:** The students are familiar with the historical development of communication technology in broad outlines.
2. **US1-25.0:** The students are able to understand the connections between computer science and other fields.
3. **US2-36.1:** The students are able to discuss computing technologies that have changed the world.

INT-009-HF:

1. **AT-27.0:** The students are able to use networks to communicate.
2. **AU-13.3:** The students are able to safely communicate ideas and information online.
3. **CH-8.0:** The students are able to use the media to maintain existing contacts and exchange information (e.g. telephone, letter).
4. **US1-1.6:** The students are able to communicate electronically with others with support student partners.

INT-010-HF:

1. **CH-25.1:** The students are able to use the media to work together.
2. **US1-2.6:** The students are able to work cooperatively and collaboratively with others using technology.

INT-011-HF:

1. **AT-44.0:** The students are able to use digital tools to collaborate.
2. **AU-5.5:** The students are able to share ideas and information with known people in safe online environments.
3. **CH-25.2:** The students are able to use the media to exchange opinions.
4. **US1-17.6:** The students are able to use productivity technology tools (e.g., word processing, spreadsheet, presentation software) for collaborative publishing activities.

INT-012-HF:

1. **AU-28.1:** The students are able to use agreed protocols when collaborating.
2. **CH-25.3:** The students are able to follow the safety rules when using media.
3. **US2-17.1:** The students are able to work respectfully with others online.

INT-013-HF:

1. **AU-28.5:** The students are able to use agreed protocols when communicating ideas, information and solutions online.
2. **US2-17.2:** The students are able to work responsibly with others online.

INT-014-HF:

1. **DE-18.0:** The students are able to name ways to protect data from unwanted access.
2. **US2-4.2:** The students are able to explain why we use passwords.

INT-015-HF:

1. **CH-21.0:** The students are able to include the security rules for handling personal data in their media content (e.g. personal details, password, nickname).
2. **GB-6.1:** The students are able to explain how to keep personal information private.

3. **US2-18.1:** The students are able to keep login information private.

INT-016-HF:

1. **AT-9.1:** The students are aware of the risks associated with the use of information technologies.
2. **GB-6.2:** The students are able to explain how to use technology safely.
3. **US2-18.2:** The students are able to log off of devices appropriately.

INT-017-HF:

1. **AT-9.2:** The students know how they should behave in a given case concerning the risks associated with the use of information technologies.
2. **GB-6.4:** The students are able to explain where to go for help and support when pupils have concerns about content.

INT-018-HF:

1. **AT-10.2:** The students are able to get help concerning the dangers of dealing with people they only know from the Internet.
2. **GB-6.5:** The students are able to explain where to go for help and support when pupils have concerns about contact.

INT-019-HF:

1. **AT-12.3:** The students are aware that there are dangers such as malware, especially when using the Internet.
2. **US2-23.1:** The students are able to discuss real-world cybersecurity problems.

INT-020-HF:

1. **AU-7.2:** The students are able to identify how digital systems are used.
2. **US2-1.3:** The students are able to recognize that users have different needs for the technology they use.

B.5 Programming and Algorithms

INT-001-PA:

1. **AT-47.1:** The students are able to understand simple instructions.
2. **CH-10.1:** The students are able to recognise formal instructions (e.g. cooking and baking recipes, play and craft instructions, dance choreographies).

INT-002-PA:

1. **AT-47.2:** The students are able to execute simple instructions.
2. **CH-10.2:** The students are able to follow formal instructions (e.g. cooking and baking recipes, play and craft instructions, dance choreographies).

INT-003-PA:

1. **AU-3.1:** The students are able to design solutions to simple problems using a sequence of steps.
2. **GB-7.6:** The students are able to solve problems by decomposing them into smaller parts.

INT-004-PA:

1. **US1-24.0:** The students are able to make a list of sub-problems to consider while addressing a larger problem.
2. **US2-11.0:** The students are able to decompose (break down) the steps needed to solve a problem into a precise sequence of instructions.

INT-005-PA:

1. **AU-10.0:** The students are able to record design ideas using techniques including labelled drawings, lists and sequenced instructions.
2. **US1-4.1:** The students are able to use writing tools to illustrate thoughts in a step-by-step manner.
3. **US1-4.2:** The students are able to use digital cameras to illustrate thoughts in a step-by-step manner.
4. **US1-4.3:** The students are able to use drawing tools to illustrate thoughts in a step-by-step manner.

5. **US1-4.4:** The students are able to use writing tools to illustrate ideas in a step-by-step manner.
6. **US1-4.5:** The students are able to use digital cameras to illustrate ideas in a step-by-step manner.
7. **US1-4.6:** The students are able to use drawing tools to illustrate ideas in a step-by-step manner.
8. **US1-4.7:** The students are able to use writing tools to illustrate stories in a step-by-step manner.
9. **US1-4.8:** The students are able to use digital cameras to illustrate stories in a step-by-step manner.
10. **US1-4.9:** The students are able to use drawing tools to illustrate stories in a step-by-step manner.

INT-006-PA:

1. **GB-1.1:** The students are able to explain what an algorithm is.
2. **US1-21.0:** The students are able to develop a simple understanding of an algorithm (e.g., search, sequence of events, or sorting) using computer-free exercises.

INT-007-PA:

1. **AU-16.2:** The students are able to design digital solutions using algorithms that involve decision-making.
2. **DE-25.4:** The students are able to design algorithms with the basic algorithmic block branching.

INT-008-PA:

1. **GB-9.2:** The students are able to use logical reasoning to detect errors in algorithms.
2. **US2-14.1:** The students are able to identify errors in an algorithm that includes sequences.
3. **US2-14.3:** The students are able to identify errors in an algorithm that includes simple loops.

INT-009-PA:

1. **GB-9.3:** The students are able to use logical reasoning to correct errors in algorithms.
2. **US2-14.2:** The students are able to fix errors in an algorithm that includes sequences.
3. **US2-14.4:** The students are able to fix errors in an algorithm that includes simple loops.

INT-010-PA:

1. **DE-27.0:** The students are able to compare algorithms using technical language.
2. **US2-26.1:** The students are able to compare multiple algorithms for the same task.

INT-011-PA:

1. **CH-36.2:** The students are able to understand that a program is a sequence of predefined instructions.
2. **GB-1.3:** The students are able to explain how programs execute by following precise and unambiguous instructions.

INT-012-PA:

1. **AU-25.0:** The students are able to plan a sequence of steps (algorithms) to create solutions, including visual programs.
2. **GB-8.1:** The students are able to use sequence in programs.
3. **US2-10.1:** The students are able to develop programs with sequences, to express ideas.
4. **US2-10.2:** The students are able to develop programs with sequences, to address a problem.

INT-013-PA:

1. **CH-37.1:** The students are able to write programs with loops.
2. **GB-8.3:** The students are able to use repetition in programs.
3. **US2-10.3:** The students are able to develop programs with simple loops, to express ideas.

4. **US2-10.4:** The students are able to develop programs with simple loops, to address a problem.

INT-014-PA:

1. **AU-16.3:** The students are able to implement digital solutions using algorithms that involve decision-making.
2. **CH-37.3:** The students are able to write programs with conditional Instructions.
3. **GB-8.2:** The students are able to use selection in programs.
4. **US2-28.4:** The students are able to create programs that include conditionals.

INT-015-PA:

1. **AU-16.5:** The students are able to implement digital solutions using algorithms that involve user input.
2. **GB-8.5:** The students are able to work with various forms of input.
3. **US2-28.2:** The students are able to create programs that include events.

INT-016-PA:

1. **GB-8.4:** The students are able to work with variables.
2. **US2-27.1:** The students are able to create programs that use variables to store data.
3. **US2-27.2:** The students are able to create programs that use variables to modify data.

INT-017-PA:

1. **GB-9.4:** The students are able to use logical reasoning to detect errors in programs.
2. **US2-14.5:** The students are able to identify errors in a program that includes sequences.
3. **US2-14.7:** The students are able to identify errors in a program that includes simple loops.

INT-018-PA:

1. **GB-9.5:** The students are able to use logical reasoning to correct errors in programs.
2. **US2-14.6:** The students are able to fix errors in a program that includes sequences.
3. **US2-14.8:** The students are able to fix errors in a program that includes simple loops.

INT-019-PA:

1. **CH-37.2:** The students are able to test programs with loops.
2. **CH-37.4:** The students are able to test programs with conditional Instructions.
3. **CH-37.6:** The students are able to test programs with Parameters.
4. **US2-33.3:** The students are able to test a program to ensure it runs as intended.

INT-020-PA:

1. **AU-40.2:** The students are able to design digital solutions that meet intended purposes including user interfaces and a visual program.
2. **GB-7.1:** The students are able to design programs that accomplish specific goals.

INT-021-PA:

1. **DE-28.0:** The students are able to program a computer science system.
2. **GB-12.4:** The students are able to create a range of systems that accomplish given goals.

C. KINDERGARTEN QUESTIONNAIRE

IT-Curriculum: Elementarbereich

Dieser Fragebogen dient dazu, das im Zuge des Projekts "Lakeside IT-Curriculum" erstellte Curriculum für den Elementarbereich zu evaluieren.

*** Erforderlich**

Bitte geben Sie an, wie viele Jahre Sie schon im Elementarbereich tätig sind. *

Meine Antwort

Haben Sie selbst schon an Aktivitäten mit Technikbezug teilgenommen? *

- ☐ Ja
- ☐ Nein

Haben Sie selbst schon Aktivitäten mit Technikbezug gestaltet? *

- ☐ Ja
- ☐ Nein

Haben Sie selbst schon an Aktivitäten mit Informatikbezug teilgenommen? *

- ☐ Ja
- ☐ Nein

Sollten Themen der Informatik bzw. digitalen Bildung für den Elementarbereich im Curriculum fehlen, tragen Sie diese bitte hier ein.

Meine Antwort

Sind Ihrer Meinung nach die im Curriculum enthaltenen Kompetenzen im Kindergarten erlernbar? *

auf jede Fall 1 2 3 4 5 nein gar nicht

☐ ☐ ☐ ☐ ☐

Geben Sie bitte ein Beispiel für eine eher unpassende Kompetenz an und begründen Sie Ihre Meinung. (Es können einfach die Nummern der Kompetenzen angegeben werden, die im Curriculum verwendet werden.) *

Meine Antwort

Geben Sie bitte ein Beispiel für eine besonders passende Kompetenz an und begründen Sie Ihre Meinung. (Es können einfach die Nummern der Kompetenzen angegeben werden, die im Curriculum verwendet werden.) *

Meine Antwort

Senden

BIBLIOGRAPHY

- [Abs07] Hermann Josef Abs. Überlegungen zur modellierung diagnostischer kompetenz bei lehrerinnen und lehrern. In M. Lüders and J. Wissinger, editors, *Forschung zur Lehrerbildung. Kompetenzenentwicklung und Programmevaluation*. Waxmann, 2007.
- [AK01] Lorin W. Anderson and David R. Krathwohl. *A taxonomy for learning, teaching, and assessing: a revision of Bloom's taxonomy of educational objectives*. Longman, New York, 2001.
- [Ang18] Renzo Angles. The property graph database model. In *Proceedings of the 12th Alberto Mendelzon International Workshop on Foundations of Data Management*, Cali, Colombia, 2018.
- [APH14] Tapio Auvinen, Juha Paavola, and Juha Hartikainen. Stops: A graph-based study planning and curriculum development tool. In *Proceedings of the 14th Koli Calling International Conference on Computing Education Research*, Koli Calling '14, pages 25–34, New York, NY, USA, 2014. ACM.
- [Aus13] Australien Curriculum. <http://www.australiancurriculum.edu.au/technologies>, 2013. Retrieved on 30.04. 2017.
- [Aus14] Austrian Agency for International Cooperation in Education and Research. The Austrian Education System. <https://www.bildungssystem.at/en/>, 2014. Retrieved on 12.08. 2019.
- [BAR12] Tim Bell, Peter Andreae, and Anthony Robins. Computer science in nz high schools: The first year of the new standards. In *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education*, SIGCSE '12, pages 343–348, New York, NY, USA, 2012. ACM.
- [BB18] Wissenschaft und Forschung (BMBWF) Bundesministerium Bildung. Digitale grundbildung. <https://www.bmbwf.gv.at/Themen/schule/zrp/dibi/dgb.html>, 2018. Retrieved on 25.03. 2020.

- [BBWV07] Klaus Berberich, Srikanta Bedathur, Gerhard Weikum, and Michalis Vazirgiannis. Comparing apples and oranges: Normalized pagerank for evolving graphs. In *WWW 2007*, pages 1145–1146. ACM, 2007.
- [BEF⁺56] B. S. Bloom, M. B. Engelhart, E. J. Furst, W. H. Hill, and D. R. Krathwohl. *Taxonomy of educational objectives. The classification of educational goals. Handbook 1: Cognitive domain*. Longmans Green, New York, 1956.
- [BEH16] M. Badawy, A. A. A. El-Aziz, and H. A. Hefny. Analysis of learning objectives for higher education textbooks using text mining. In *2016 12th International Computer Engineering Conference (ICENCO)*, pages 202–207, Dec 2016.
- [Bel14] Timothy C. Bell. Establishing a nationwide cs curriculum in new zealand high schools. *Commun. ACM*, 57:28–30, 2014.
- [Ber13] Miles Berry. *Computing in the national curriculum - A guide for primary teachers*. Computing at School, 2013.
- [Ber15] Miles Berry. *QuickStart Primary Handbook*. Swindon: BCS, 2015.
- [BJG07] Jorgen Bang-Jensen and Gregory Gutin. *Digraphs: Theory, Algorithms and Applications*. Springer, 2007.
- [BMD⁺15] Erik Barendsen, Linda Mannila, Barbara Demo, Nataša Grgurina, Cruz Izu, Claudio Mirolo, Sue Sentance, Amber Settle, and Gabrièle Stupurienė. Concepts in k-9 computer science education. In *Proceedings of the 2015 ITiCSE on Working Group Reports*, ITiCSE-WGR '15, pages 85–116, New York, NY, USA, 2015. ACM.
- [BNJ03] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3(null):993–1022, March 2003.
- [Bru60] Jerome S. Bruner. *The process of education*. Cambridge Mass, 1960.
- [Bru10] Richard A. Brualdi. *Introductory Combinatorics*. Pearson Education, New York, 5 edition, 2010.
- [Bru14] Rik Van Bruggen. *Learning Neo4j*. Packt Publishing, 2014.
- [BSCH14] Neil C. C. Brown, Sue Sentance, Tom Crick, and Simon Humphreys. Restart: The resurgence of computer science in uk schools. *Trans. Comput. Educ.*, 14(2):9:1–9:22, June 2014.

- [Bun18a] Bundesinstitut für Bildungsforschung, Innovation und Entwicklung des österreichischen Schulwesens (BIFIE). Kompetenzmodelle und Bildungsstandards. <https://www.bifie.at/kompetenzmodelle-und-bildungsstandards/>, 2018. Retrieved on 05.12. 2018.
- [Bun18b] Bundesministerium Bildung, Wissenschaft und Forschung (BMBWF). Bildungsstandards in der Allgemeinbildung. <https://bildung.bmbwf.gv.at/schulen/unterricht/ba/bildungsstandards.html>, 2018. Retrieved on 05.12. 2018.
- [Bur16] Diane M. Burnette. The renewal of competency-based education: A review of the literature. *The Journal of Continuing Higher Education*, 64(2):84–93, 2016.
- [BWD00] Moshe Barak, Shlomo Waks, and Yaron Doppelt. Majoring in technology studies at high school and fostering learning. *Learning Environments Research*, 3(2):135–158, May 2000.
- [Car93] David Carr. Question of competence. *British Journal of Educational Studies*, 41(3):253–271, 1993.
- [Chy19] Yelyzaveta Chystoplova. Determination of semantic similarity between competencies from computer science curricula. Master’s thesis, National Technical University Kharkiv Polytechnic Institute, Kharkiv, Ukraine, 2019.
- [Chy20] Yelyzaveta Chystoplova. An ontological approach for learning outcomes classification in computer science. Master’s thesis, National Technical University Kharkiv Polytechnic Institute, Kharkiv, Ukraine, 2020. to be published.
- [CL06] Chengling Zhao and Liyong Wan. A shortest learning path selection algorithm in e-learning. In *Sixth IEEE International Conference on Advanced Learning Technologies (ICALT’06)*, pages 94–95, July 2006.
- [CMM18] Louis Cohen, Lawrence Manion, and Keith Morrison. *Research methods in education*. Routledge, London, New York, 8 edition, 2018.
- [Cod80] E. F. Codd. Data models in database management. *SIGPLAN Not.*, 16(1):112–114, June 1980.

- [Com12] Computing at School Working Group. Computer Science: A curriculum for schools. <https://www.computingatschool.org.uk/data/uploads/ComputingCurric.pdf>, 2012. Retrieved on 30.04. 2017.
- [CST16] CSTA (Computer Science Teachers Association). K-12 Computer Science Framework. Retrieved from <http://www.k12cs.org>, 2016. Retrieved on 04.09. 2019.
- [CST17] CSTA (Computer Science Teachers Association). K-12 Computer Science Standards, Revised 2017. Retrieved from <http://www.csteachers.org/standards>, 2017. Retrieved on 30.04. 2017.
- [CVP17] Stephanie Carretero, Rina Vuorikari, and Yves Punie. Digcomp 2.1: The digital competence framework for citizens with eight proficiency levels and examples of use. 2017. Retrieved on 09.08. 2019.
- [DB15] Caitlin Duncan and Tim Bell. A pilot computer science and programming course for primary school students. In *Proceedings of the Workshop in Primary and Secondary Computing Education*, WiP-SCE '15, pages 39–48, New York, NY, USA, 2015. ACM.
- [Die16] Reinhard Diestel. *Graph Theory*. Springer, New York, USA, electronic edition 5 edition, 2016.
- [Dig13a] Digikomp Initiative. Digikomp12 competency model. <https://digikomp.at/index.php?id=585&L=0>, 2013. Retrieved on 14.08. 2019.
- [Dig13b] Digikomp Initiative. Digikomp4 competency model. <https://digikomp.at/index.php?id=542&L=0>, 2013. Retrieved on 14.08. 2019.
- [Dig13c] Digikomp Initiative. Digikomp8 competency model. <https://digikomp.at/index.php?id=557&L=0>, 2013. Retrieved on 14.08. 2019.
- [Dij59] Edsger W Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959.
- [dkEE18] Schweizerische Konferenz der kantonalen Erziehungsdirektoren (EDK). Nationale Bildungsziele. <http://www.edk.ch/dyn/12930.php>, 2018. Retrieved on 04.12. 2018.

- [Dör10] Christina Dörge. Competencies and skills: Filling old skins with new wine. In Nicholas Reynolds and Márta Turcsányi-Szabó, editors, *Key Competencies in the Knowledge Society*, pages 78–89, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [Dub98] David D. Dubois. *The Competency Casebook*. HRD Press, 1998.
- [Duc02] Charles Duchâteau. *Information and communication technology in education: a curriculum for schools and programme of teacher development*. UNESCO, 2002. Publication code : **RES. ACAD.
- [EoIE16] Joint Informatics Europe and ACM Europe Working Group on Informatics Education 2013. Informatics Education: Europe cannot afford to miss the boat: Report of the joint Informatics Europe and ACM Europe Working Group on Informatics Education. [Online] <http://europe.acm.org/iereport/ACMandIEreport.pdf>, [Accessed: May. 6th, 2016].
- [Eur18] European Commission/EACEA/Eurydice. The Structure of the European Education Systems 2018/19: Schematic Diagrams. Eurydice Facts and Figures. https://eacea.ec.europa.eu/national-policies/eurydice/sites/eurydice/files/the_structure_of_the_european_education_systems_2018_19.pdf, 2018. Retrieved on 12.08. 2019.
- [Fer13] Ferrari, Anusca and Brecko, Barbara and Punie, Yves. DIGCOMP: A Framework for Developing and Understanding Digital Competence in Europe (Report EUR 26035 EN), 2013.
- [Fle81] Joseph L. Fleiss. *Statistical methods for rates and proportions*. John Wiley, New York, NY, USA, 2 edition, 1981.
- [FVF14] Katrina Falkner, Rebecca Vivian, and Nickolas Falkner. The australian digital technologies curriculum: Challenge and opportunity. In *Proceedings of the Sixteenth Australasian Computing Education Conference - Volume 148*, ACE '14, pages 3–12, Darlinghurst, Australia, Australia, 2014. Australian Computer Society, Inc.
- [GBWB15] Allan A. Glatthorn, Floyd Boschee, Bruce M. Whitehead, and Bonni F. Boschee. *Curriculum Leadership - Strategies for Development and Implementation*. Sage Publications Inc., California, USA, 4 edition, 2015.

- [Ges08] Gesellschaft für Informatik (GI). Bildungsstandards Informatik für die Sekundarstufe I. https://www.gi.de/fileadmin/redaktion/empfehlungen/Bildungsstandards_2008.pdf, 2008. Retrieved on 30.04. 2017.
- [Ges16] Gesellschaft für Informatik (GI). Bildungsstandards Informatik für die Sekundarstufe II. <https://www.gi.de/fileadmin/redaktion/empfehlungen/Bildungsstandards-Informatik-SekII.pdf>, 2016. Retrieved on 30.04. 2017.
- [Ges19] Gesellschaft für Informatik (GI). Kompetenzen für informatische Bildung im Primarbereich, 2019. Attachment.
- [Gua17] Guan, Jing and Loo, Bryce and Trines, Stefan. Education in Australia. <https://wenr.wes.org/2017/12/education-in-australia>, 2017. Retrieved on 12.08. 2019.
- [HAB⁺11] Peter Hubwieser, Michal Armoni, Torsten Brinda, Valentina Dagiene, Ira Diethelm, Michail N. Giannakos, Maria Knobelsdorf, Johannes Magenheimer, Roland Mittermeir, and Sigrid Schubert. Computer science/informatics in secondary education. In *Proceedings of the 16th Annual Conference Reports on Innovation and Technology in Computer Science Education - Working Group Reports*, ITiCSE-WGR '11, pages 19–38, New York, NY, USA, 2011. ACM.
- [Hel07] Silke Hellwig. *Zur Vereinbarkeit von Competency-Based-Training (CBT) und Berufsprinzip : Konzepte der Berufsbildung im Vergleich*. VS, Verl. für Sozialwiss Wiesbaden, 1. aufl. edition, 2007.
- [Her13] Walter Herzog. *Bildungsstandards (Praxiswissen Bildung)*. Kohlhammer W. Verlag, 1. aufl. edition, 2013.
- [HGB⁺15] Peter Hubwieser, Michail N. Giannakos, Marc Berges, Torsten Brinda, Ira Diethelm, Johannes Magenheimer, Yogendra Pal, Jana Jackova, and Egle Jasute. A global snapshot of computer science education in k-12 schools. In *Proceedings of the 2015 ITiCSE on Working Group Reports*, ITiCSE-WGR '15, pages 65–83, New York, NY, USA, 2015. ACM.
- [HJ04] Raymond A. Horn Jr. *Standards*. Peter Lang, New York, USA, 2004.

- [HKP12] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, San Francisco, CA, USA, 3 edition, 2012.
- [Hub08] Peter Hubwieser. Analysis of learning objectives in object oriented programming. In *Proceedings of the 3rd International Conference on Informatics in Secondary Schools - Evolution and Perspectives: Informatics Education - Supporting Computational Thinking*, ISSEP '08, pages 142–150, Berlin, Heidelberg, 2008. Springer-Verlag.
- [Jih96] Hueyching Janice Jih. The impact of learners' pathways on learning performance in multimedia computer aided learning. *Journal of Network and Computer Applications*, 19(4):367 – 380, 1996.
- [JTFoCCS13a] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. ACM, New York, NY, USA, 2013.
- [JTFoCCS13b] Association for Computing Machinery (ACM) Joint Task Force on Computing Curricula and IEEE Computer Society. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. Association for Computing Machinery, New York, NY, USA, 2013.
- [JVP02] Elizabeth A. Jones, Richard A. Voorhees, and Karen Paulson. Defining and assessing learning: Exploring competency-based initiatives. 2002.
- [KAB⁺04] Eckhard Klieme, Hermann Avenarius, Werner Blum, Peter D'öbrich, Hans Gruber, Manfred Prenzel, Kristina Reiss, Kurt Riquarts, Jürgen Rost, Heinz-Elmar Tenorth, and Helmut J. Vollmer. The development of national educational standards - an expertise. 2004.
- [KAB⁺07] Eckhard Klieme, Hermann Avenarius, Werner Blum, Peter D'öbrich, Hans Gruber, Manfred Prenzel, Kristina Reiss, Kurt Riquarts, Jürgen Rost, Heinz-Elmar Tenorth, and Helmut J. Vollmer. Zur entwicklung nationaler bildungsstandards. 2007.
- [KC12] Rebecca Klein-Collins. Competency-based degree programs in the u.s.: Postsecondary credentials for measurable student learning and performance. 2012.

- [Kel04] Albert Victor Kelly. *The Curriculum: Theory and Practice*. Sage Publications Ltd., London, England, 6 edition, 2004.
- [KHR08] Eckhard Klieme, Johannes Hartig, and Dominique Rauch. The concept of competence in educational contexts. In Johannes Hartig, Eckhard Klieme, and Detlev Leutner, editors, *Assessment of Competencies in Educational Contexts*. Hogrefe and Huber Publishers, 2008.
- [KL06] Eckhard Klieme and Detlev Leutner. Kompetenzmodelle zur erfassung individueller lernergebnisse und zur bilanzierung von bildungsprozessen. beschreibung eines neu eingerichteten schwerpunktprogramms der dfg. *Zeitschrift für Pädagogik*, (52):876–903, 2006.
- [Kli04] Eckhard Klieme. Was sind kompetenzen und wie lassen sie sich messen? *Pädagogik*, (56):10–13, 2004.
- [KMK18] Kultusministerkonferenz KMK. Bildungsstandards der Kultusministerkonferenz. <https://www.kmk.org/themen/qualitaetssicherung-in-schulen/bildungsstandards.html>, 2018. Retrieved on 04.12. 2018.
- [KMM08] Eckhard Klieme and Katharina Maag-Merki. Introduction of educational standards in german-speaking countries. In Johannes Hartig, Eckhard Klieme, and Detlev Leutner, editors, *Assessment of Competencies in Educational Contexts*. Hogrefe and Huber Publishers, 2008.
- [KMMH07] Eckhard Klieme, Katharina Maag-Merki, and Johannes Hartig. Kompetenzbegriff und bedeutung von kompetenzen im bildungswesen. *M'oglichkeiten und Voraussetzungen technologiebasierter Kompetenzdiagnostik*, 2007.
- [Lam16] Phil Lambert. Educational Standards and Australia: a changed landscape. *Rev. Bras. Estud. Pedagog.* [online], 97(247):463–471, 2016.
- [Leh14] Lehrplan 21. <http://v-ef.lehrplan.ch>, 2014. Retrieved on 30.04. 2017.
- [Lig10] Jay M. Lightfoot. A Graph-Theoretic Approach to Improved Curriculum Structure and Assessment Placement. *Communications of the IIMA*, 10(2):59–73, 2010.

- [Loo18] Loo, Bryce. Education in Australia. <https://wenr.wes.org/2018/06/education-in-the-united-states-of-america>, 2018. Retrieved on 12.08. 2019.
- [LWS14] Cecilia Le, Rebecca E. Wolfe, and Adria Steinberg. The past and the promise: Today’s competency education movement. 2014.
- [Mar12] Linda Marshall. A comparison of the core aspects of the acm/ieee computer science curriculum 2013 strawman report with the specified core of cc2001 and cs2008 review. In *Proceedings of Second Computer Science Education Research Conference, CSERC ’12*, pages 29–34, New York, NY, USA, 2012. ACM.
- [Mar14] Linda Marshall. *A graph-based framework for comparing curricula*. PhD thesis, University of Pretoria - Department of Computer Science, 2014.
- [May15] Philipp Mayring. *Qualitative Inhaltsanalyse : Grundlagen und Techniken*. Weinheim : Basel : Beltz, 12 edition, 2015.
- [McM08] James H. McMillan. *Assessment Essentials for Standards-Based Education*. Corwin Press, California, USA, 2 edition, 2008.
- [MDE13] Anne Mette Morcke, Tim Dornan, and Berit Eika. Outcome (competency) based education: an exploration of its origins, theoretical basis, and empirical evidence. *Advances in Health Sciences Education*, 18(4):851–863, 10 2013.
- [Mes06] Helmut Messner. Über das offene verhältnis von inhalten und ziele der didaktik. In Lucien Criblez, Peter Gautschi, Pia Hirt Monico, and Helmut Messner, editors, *Lehrpläne und Bildungsstandards. Was Schülerinnen und Schüler lernen sollen*. h.e.p. Verlag, Bern, Switzerland, 2006.
- [Mey06] Bertrand Meyer. Testable, reusable units of cognition. *Computer*, 39(4):20–24, April 2006.
- [MG06] Allan Martin and Jan Grudziecki. Digeulit: Concepts and tools for digital literacy development. *Innovation in Teaching and Learning in Information and Computer Sciences*, 5(4):249–267, 2006.
- [Moe20] Philipp Moedritscher. Gecko upgrade documentation. Technical report, 2020. (unpublished development report).

- [MPB17] Peter Micheuz, Stefan Pasterk, and Andreas Bollin. Basic digital education in austria – one step further. In Arthur Tatnall and Mary Webb, editors, *Tomorrow's Learning: Involving Everyone. Learning with and about Technologies and Computing*, pages 432–442, Cham, 2017. Springer International Publishing.
- [MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, USA, 2008.
- [MZ13] Ursula Mulley and Barbara Zuliani. Digitales Kompetenzmodell für die Volksschule. *Digitale Schule Österreich, OCG Schriftenreihe*, 2013.
- [Nat14] National Curriculum UK. <https://www.gov.uk/government/collections/national-curriculum>, 2014. Retrieved on 30.04. 2017.
- [Neo] Neo4j Graph Database. <https://neo4j.com/>. Retrieved on 30.04. 2017.
- [New07] M. E. J. Newman. Mathematics of networks. In Steven N. Durlauf and Lawrence E. Blume, editors, *The New Palgrave Dictionary of Economics*. Palgrave Macmillan, 2 edition, 2007.
- [NG96] Hans Niedderer and Fred Goldberg. Learning processes in electric circuits. In *NARST Annual Meeting*. St.Louis, Missouri, USA, 1996.
- [NH19] Mark Needham and Amy E. Hodler. *Graph Algorithms: Practical Examples in Apache Spark and Neo4j*. O'Reilly Media, Inc., Sebastopol, CA, USA, 2019.
- [oBC13] Government of British Columbia. Digital literacy framework. <https://www2.gov.bc.ca/gov/content/education-training/k-12/teach/teaching-tools/digital-literacy>, 2013. Retrieved on 09.01. 2018.
- [oECEC17] The Committee on European Computing Education (CECE). Informatics education in europe: Are we all in the same boat? Technical report, New York, NY, USA, 2017.
- [oIT] Bebras International Challenge on Informatics and Computational Thinking. <https://www.bebas.org/>. Retrieved on 27.09. 2019.

- [otEU06] Council of the European Union. Council recommendation of 18 december 2006 on key competences for lifelong learning. 2006. Retrieved on 08.08. 2019.
- [otEU18] Council of the European Union. Council recommendation of 22 may 2018 on key competences for lifelong learning. 2018. Retrieved on 08.08. 2019.
- [PB17a] S. Pasterk and A. Bollin. Digital literacy or computer science: Where do information technology related primary education models focus on? In *2017 15th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, pages 1–7, Oct 2017.
- [PB17b] S. Pasterk and A. Bollin. Graph-based analysis of computer science curricula for primary education. In *2017 IEEE Frontiers in Education Conference (FIE)*, pages 1–9, Oct 2017.
- [PK18] Stefan Pasterk and Max Kesselbacher. (semi-)automatically processing and analyzing computer science curricula with nlp. Technical report, 2018. (unpublished seminar paper).
- [PKB19] Stefan Pasterk, Max Kesselbacher, and Andreas Bollin. A semi-automated approach to categorise learning outcomes into digital literacy or computer science. In Don Passey, Rosa Bottino, Cathy Lewin, and Eric Sanchez, editors, *Empowering Learners for Life in the Digital Age*, pages 77–87, Cham, 2019. Springer International Publishing.
- [PM10] Michela Pedroni and Bertrand Meyer. Object-oriented modeling of object-oriented concepts. In Juraj Hromkovič, Richard Kráľovič, and Jan Vahrenhold, editors, *Teaching Fundamentals Concepts of Informatics*, pages 155–169, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.
- [POM07] Michela Pedroni, Manuel Oriol, and Bertrand Meyer. A framework for describing and comparing courses and curricula. In *Proceedings of the 12th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education*, ITiCSE ’07, pages 131–135, New York, NY, USA, 2007. ACM.
- [Pre12] Christina Prell. *Social network analysis: history, theory and methodology*. SAGE, London, UK, 2012.

- [Rav95] Diane Ravitch. *National Standards in American Education: A Citizen's Guide*. Brookings Institution Press, Washington D.C., USA, 1995.
- [RN10] Marko A. Rodriguez and Peter Neubauer. Constructions from dots and lines. *CoRR*, abs/1006.2361, 2010.
- [RWE15] Ian Robinson, Jim Webber, and Emil Eifrem. *Graph Databases*. O'Reilly, Sebastopol, CA, USA, 2 edition, 2015.
- [SCF⁺11] Deborah Seehorn, Stephen Carey, Brian Fuschetto, Irene Lee, Daniel Moix, Dianne O'Grady-Cunniff, Barbara Boucher Owens, Chris Stephenson, and Anita Verno. Csta k-12 computer science standards: Revised 2011. Technical report, New York, NY, USA, 2011. 104111.
- [Sch94] Andreas Schwill. Fundamental ideas in computer science. *Bulletin European Association for Theoretical Computer Science*, (53):274–295, 1994.
- [Sch15] European Schoolnet. Computing our future: Computer programming and coding? priorities, school curricula and initiatives across europe (update 2015. Technical report, 2015. Update 2015.
- [SGGA⁺10] J. A. Sainz, J. M. Gil-García, L. A. Aguado, A. Aledo, and J. Quesada. Knowledge management and professional profiles in electronic systems engineering: The function of university-industry collaboration. In *IEEE EDUCON 2010 Conference*, pages 365–368, April 2010.
- [SHLA18] Maria Spante, Sylvana Sofkova Hashemi, Mona Lundin, and Anne Algers. Digital competence and digital literacy in higher education research: Systematic review of concept use. *Cogent Education*, 5(1):1519143, 2018.
- [SK15] Maciej M. Sysło and Anna Beata Kwiatkowska. Introducing a new computer science curriculum for all school levels in poland. In Andrej Brodnik and Jan Vahrenhold, editors, *Informatics in Schools. Curricula, Competences, and Competitions*, pages 141–154, Cham, 2015. Springer International Publishing.
- [SMY10] Takayuki Sekiya, Yoshitatsu Matsuda, and Kazunori Yamaguchi. Analysis of computer science related curriculum on lda and isomap. In *Proceedings of the Fifteenth Annual Conference on Innovation*

- and Technology in Computer Science Education*, ITiCSE '10, pages 48–52, New York, NY, USA, 2010. ACM.
- [Soc] The Royal Society. Shut down or restart? the way forward for computing in uk schools.
- [SP06] Horst Schecker and Ilka Parchmann. Modellierung naturwissenschaftlicher kompetenz. *Zeitschrift für Didaktik der Naturwissenschaften*, (12):45–66, 2006.
- [Spa94] William G. Spady. *Outcome-based education: Critical issues and answers*. American Association of School Administrators, USA, 1994.
- [Spa20] Spacy Usage Documentation. <https://spacy.io/usage/>, 2020. Retrieved on 27.02. 2020.
- [SSST09] Marc S. Schwartz, Philip M. Sadler, Gerhard Sonnert, and Robert H. Tai. Depth versus breadth: How content coverage in high school science courses relates to later success in college science coursework. *Science Education*, 93(5):798–826, 2009.
- [Ste10] Markus Steinert. *Lernzielstrukturen im Informatikunterricht*. habilitation, Technischen Universität München, 2010.
- [SW12] Daniel Sundberg and Ninni Wahlstrom. Standards-based curricula in a denationalised conception of education: The case of sweden. *European Educational Research Journal*, 11(3):342–356, 2012.
- [TD17] Andrey Tkachuk and Vadym Demchuk. Gecko documentation. Technical report, 2017. (unpublished development report).
- [Teo06] Tina Teodorescu. Competence versus competency: What is the difference? *Performance Improvement*, 45(10):27–30, 2006.
- [TJTfCCS01] Association for Computing Machinery (ACM) The Joint Task Force on Computing Curricula and IEEE Computer Society. Computing curricula 2001. *J. Educ. Resour. Comput.*, 1(3es):1–es, September 2001.
- [TJTfCCS08] Association for Computing Machinery (ACM) The Joint Task Force on Computing Curricula and IEEE Computer Society. Computing curricula 2008: An interim revision of cs 2001. 2008.

- [TKMS03] Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, page 173–180, USA, 2003. Association for Computational Linguistics.
- [Tuc03] Allen Tucker. A model curriculum for k–12 computer science: Final report of the acm k–12 task force curriculum committee. Technical report, New York, NY, USA, 2003. ACM Order No.: 104043.
- [VdHM97] Helen Van der Horst and Ria McDonald. *OBE (Outcome-based Education): A Teacher's Manual*. Kagiso Publishers, Pretoria, 1997.
- [Ver19] Marharyta Verkhovets. Standardization of computer science curricula based on selection of semantically close concepts. Master's thesis, National Technical University Kharkiv Polytechnic Institute, Kharkiv, Ukraine, 2019.
- [WBD⁺18] Mary Webb, Timothy Bell, Niki Davis, Yaacov Katz, Andrew Fluck, Maciej Syslo, Ivan Kalas, Margaret Cox, Charoula Angeli, Joyce Malyn-Smith, Torsten Brinda, Peter Micheuz, and Andrej Brodnik. Tensions in specifying computing curricula for k-12: Towards a principled approach for objectives. *it - Information Technology*, 60:59–68, 04 2018.
- [WDB⁺17] Mary Webb, Niki Davis, Tim Bell, Yaacov J. Katz, Nicholas Reynolds, Dianne P. Chambers, and Maciej M. Sysło. Computer science in k-12 school curricula of the 21st century: Why, what and when? *Education and Information Technologies*, 22(2):445–468, Mar 2017.
- [Wei99] Franz E. Weinert. Concepts of competence. 1999.
- [Wei01a] Franz E. Weinert. Concept of competence: a conceptual clarification. In D.S. Rychen and L.H. Salganik, editors, *Defining and selecting key competencies*, pages 45–65. Hogrefe and Huber Publishers, Seattle, 2001.
- [Wei01b] Franz E. Weinert. Vergleichende leistungsmessung in schulen – eine umstrittene selbstverständlichkeit. In Franz E. Weinert, editor, *Leistungsmessungen in Schulen*, pages 17–31. Beltz Verlag, Weinheim, 2001.

-
- [Win10] Esther Winther. *Kompetenzmessung in der beruflichen Bildung*. W. Bertelsmann Verlag, San Francisco, CA, 2010.
- [WSBP17] Christian Wiesner, Claudia Schreiner, Simone Breit, and Katrin Pacher. *Bildungsstandards und kompetenzorientierter unterricht*. 2017.

LIST OF FIGURES

Chapter 2

2.1 Hierarchical relationship of skills, knowledge and competencies [JVP02, p. 8]	19
---	----

Chapter 3

3.1 Overview of the educational systems of selected countries	46
3.2 Content and Process areas of the GI Standards [Ges08, Ges19]	52
3.3 Comparison of covered age-ranges of selected curricula	57

Chapter 4

4.1 Graphical representation of graph G	64
4.2 Graphical representation of digraph D	66
4.3 The first rows for the first two levels of the progression chart for the CSTA Standards 2017 [CST17]	76
4.4 The first rows for the third and fourth level of the progression chart for the CSTA Standards 2017 [CST17]	77
4.5 Transitivity in the case of a first <i>requires</i> -relation followed by an <i>expands</i> -relation	78
4.6 Transitivity in the case of a first <i>expands</i> -relation followed by an <i>requires</i> -relation	79
4.7 An example of a property graph using the properties for competencies and the two relation-types	81
4.8 The graph schema for the competency graphs	83
4.9 CSTA 2011 category 'Collaboration' showing dependencies within the category	87
4.10 CSTA 2011 category 'Collaboration' including dependencies to other categories	88

Chapter 5

5.1 Comparison of the subcompetencies per competency cluster	114
5.2 Comparison of the absolute numbers of original competencies and standardized competencies	115

5.3	Comparison of the absolute numbers of relations in the original curricula	116
5.4	Comparison of the number of relations per original competency . . .	116
5.5	Comparison of the absolute numbers of relations in the standardized curricula	117
5.6	Comparison of the number of relations per standardized competency .	118
5.7	Comparison of the relations per original and per standardized competency	119
5.8	Comparison of the density values for both relation types between original competencies	120
5.9	Comparison of the density values for both relation types between standardized competencies	121
5.10	Comparison of the degree values in the standardized curricula	129
5.11	Comparison of the relative number of sources, sinks and isolated nodes in the original curricula	136
5.12	Comparison of the relative number of sources, sinks and isolated nodes in the standardized curricula	138
5.13	Comparison of normalized numbers of connected components between original and standardized curricula	141
5.14	Comparison of the focus from the five analyzed curricula	142
5.15	Comparison of the strong agreements in the experts' decisions	143
5.16	The distribution of experts' choices for CS for the learning objectives from the Australian curriculum [PB17a]	145
5.17	The distribution of experts' choices for DL for the learning objectives from the Australian curriculum [PB17a]	145
5.18	A comparison of learning objectives related to <i>CS</i> or <i>DL</i> from the Australian curriculum [PB17a]	146
5.19	The distribution of experts' choices for CS for the competence levels from the curriculum in Switzerland [PB17a]	147
5.20	The distribution of experts' choices for DL for the competence levels from the curriculum in Switzerland [PB17a]	148
5.21	A comparison of competence levels related to <i>CS</i> or <i>DL</i> from the curriculum from Switzerland [PB17a]	149
5.22	The distribution of experts' choices for CS for the competences from the Austrian competence model [PB17a]	150
5.23	The distribution of experts' choices for DL for the competences from the Austrian competence model [PB17a]	151
5.24	A comparison of competences related to <i>CS</i> or <i>DL</i> from the Austrian competence model [PB17a]	151
5.25	Relative distribution of original competencies over the five categories	152
5.26	Comparison of the five categories based on original competencies . . .	152

5.27 Relative distribution of standardized competencies over the five categories	153
--	-----

Chapter 6

6.1 The <i>GGBMC</i> in a multicircle layout	156
6.2 Comparison of relative numbers of sinks, sources and isolated nodes .	158
6.3 Numbers of competencies in each curriculum combined to intersector nodes	159
6.4 Comparison of numbers of competencies and intersector nodes in the categories	159
6.5 Comparison of normalized degree, in-degree, and out-degree values .	162
6.6 Prerequisites graph for the node US1-24.0	172
6.7 Prerequisites graph for the node INT-004-PA	174

Chapter 7

7.1 Screenshot of the GECKO web platform	183
7.2 Screenshot of the GECKO web platform showing the competency information bar	184
7.3 The competency information bar showing a learning path for the selected node to a sink	185
7.4 A part of the relation information bar	186
7.5 The competency information bar with the possibility to evaluate a node	188
7.6 The competency information bar with the possibility to evaluate a relation	189
7.7 Statistical overview of the number of evaluated elements and some results	189
7.8 A part of the overview of the evaluation results	190
7.9 The architecture of the GECKO system [TD17]	191
7.10 The Entity Relationship Diagram for the GECKO system [Moe20] .	193

Chapter 8

8.1 Relative <i>df</i> of the five most frequently used nouns (adapted from [PK18]).	211
8.2 Mean <i>tf-idf</i> value of the five most frequently used nouns for each curriculum (adapted from [PK18]).	212
8.3 Relative <i>df</i> of the five most frequently used verbs (adapted from [PK18]).	214
8.4 Mean <i>tf-idf</i> value of the five most frequently used verbs for each curriculum (adapted from [PK18]).	214
8.5 Occurrence of nouns in the categories (adapted from [Chy20]). . . .	216
8.6 Occurrence of unique nouns in the categories (adapted from [Chy20]).	217

8.7	Example for learning outcome processing from CSTA standards from 2011 [PKB19].	229
8.8	Percentage of matches in the categories	234
8.9	Percentage of matches in the curricula	234
8.10	Percentage of matches in the categories within the curricula	235

Chapter 9

9.1	Table representation of all competencies for discussion.	244
9.2	Table extract showing three similar competencies.	245
9.3	Dependencies structure for all selected competencies	247
9.4	Simple dependency structure for one <i>target competency</i> and one <i>support competency</i>	249
9.5	More complex dependency structure for one <i>target competency</i> and one <i>support competency</i>	250
9.6	Complex dependency structure for two <i>target competency</i> and one <i>support competency</i>	251
9.7	Matches of automatically determined and from experts selected learning paths	259

LIST OF TABLES

Chapter 2

2.1	Characteristics of the different learning objective levels [AK01]	25
2.2	The cognitive process dimension adapted from [AK01]	28
2.3	The Taxonomy Table [AK01]	29

Chapter 3

3.1	Knowledge categories and referred knowledge areas from the ACM/IEEE Curriculum [BMD ⁺ 15]	37
3.2	Relative distribution of the knowledge-category-code occurrences within the curricula [BMD ⁺ 15]	39
3.3	Comparison of six countries [WBD ⁺ 18]	41
3.4	Comparison of relevant Education Systems	45
3.5	Comparison of duration of the selected educational models	58
3.6	Comparison of the absolute numbers of learning objectives	58
3.7	Comparison of the numbers of learning objectives relative to the covered years	59

Chapter 5

5.1	Numbers of original competencies and standardized subcompetencies	113
5.2	Numbers of original competencies and standardized subcompetencies	120
5.3	Maximum values of degree (including in- and out-degree) in the original and standardized curricula	122
5.4	Comparison of the identified competencies by different centrality measures (the colors highlight competencies reappearing in different measures)	133
5.5	Numbers of sources, sinks and isolated nodes in the original and standardized curricula	135
5.6	Numbers of connected components	139
5.7	Percentage of the categories <i>computer science</i> and <i>digital literacy</i> in curricula	143

Chapter 6

6.1	Results for basic measures of the GGBMC	157
6.2	Comparison of the identified nodes by different centrality measures in the GGBMC (the colors highlight nodes which appear in different measures)	165

Chapter 8

8.1	Vector for the two competencies CH-37.1 and US2-10.3 using term frequency	206
8.2	Basic numbers of the formulations in the curricula (adapted from [PK18]).	208
8.3	Overview of the top 15% terms (measured with cumulative <i>tf-idf</i> scores) of the analyzed Australian curriculum (adapted from [PK18]).	210
8.4	Comparison of the five most frequent nouns in each curriculum (adapted from [PK18]).	210
8.5	Comparison of the five most frequent verbs in each curriculum (adapted from [PK18]).	213
8.6	Comparison of the five most frequent adjectives and adverbs in each curriculum (adapted from [PK18]).	215
8.7	A contingency table to calculate <i>precision</i> and <i>recall</i> [MRS08, p. 155].	221
8.8	The contingency table for the CSTA11 standards using <i>Jaccard coefficient</i> and <i>cosine similarity</i> with <i>nouns and verbs</i>	222
8.9	Weights for the different cases of occurring POS elements [Chy19]. . .	224
8.10	The contingency table for the CSTA11 standards using modified <i>Jaccard coefficient</i> with <i>nouns and verbs</i>	224
8.11	The contingency table for the overall results from modified <i>Jaccard coefficient</i> with <i>nouns and verbs</i>	226
8.12	<i>Precision</i> and <i>recall</i> considering only <i>expands</i> relations.	227
8.13	Comparison of results (adapted from [PKB19])	231
8.14	Application of categorization with two different dictionaries (adapted from [PKB19])	232

Chapter 9

9.1	The results from the survey with kindergarten educators.	255
-----	--	-----

LIST OF LISTINGS

Chapter 4

4.1 Query for the density of a graph	85
--	----

Chapter 7

7.1 Query for the PageRank in single curricula for the <i>expands</i> relations [Moe20]	194
7.2 Query for the PageRank in single curricula [Moe20]	195
7.3 Query for the PageRank including intersector nodes [Moe20]	196
7.4 Query for shortest path determination [Moe20]	197
7.5 Query for learning paths with high overall centrality [Moe20]	198

Chapter 8

8.1 Python function used for tokenization, tagging and lemmatizing [PK18].	203
8.2 SpaCy library used for tokenization, tagging and lemmatizing	204
8.3 Python function to compute the tf-idf measures for the learning out- comes of an unstructured text file using NLTK library [PK18].	209
8.4 Example of a dictionary entry for <i>nouns</i> [Ver19].	218
8.5 Example of a dictionary entry for <i>verbs</i> [Ver19].	219

LIST OF DEFINITIONS

Chapter 2

2.1	Definition (Curriculum 1)	11
2.2	Definition (Curriculum 2)	12
2.3	Definition (Outcome-based education)	13
2.4	Definition (Outcome)	13
2.5	Definition (Standards 1)	13
2.6	Definition (Standards 2)	13
2.7	Definition (Standards 3)	15
2.8	Definition (Competence 1)	18
2.9	Definition (Competency 1)	18
2.10	Definition (Competency 2)	20
2.11	Definition (Knowledge 1)	21
2.12	Definition (Skill 1)	21
2.13	Definition (Meta-knowledge)	21
2.14	Definition (Knowledge 2)	21
2.15	Definition (Skill 2)	21
2.16	Definition (Attitude)	21
2.17	Definition (Competency 3)	22
2.18	Definition (Subcompetency)	22
2.19	Definition (Informatics and Computing Science)	30
2.20	Definition (Informatics technology)	30
2.21	Definition (Information and communication technology (ICT))	30
2.22	Definition (Computer Science and Informatics)	31
2.23	Definition (Digital literacy)	32
2.24	Definition (Digital competence)	33

Chapter 4

4.1	Definition (Graph)	63
4.2	Definition (Loop)	64
4.3	Definition (Subgraph)	64
4.4	Definition (Degree)	65

4.5	Definition (Directed-graph)	65
4.6	Definition (Directed-pseudograph)	66
4.7	Definition (Size)	66
4.8	Definition (In- and out-degree)	67
4.9	Definition (Subdigraph)	67
4.10	Definition (Walk)	67
4.11	Definition (Path)	67
4.12	Definition (Cycle)	67
4.13	Definition (Acyclic-digraph)	68
4.14	Definition (Biorientation)	68
4.15	Definition (Underlying graph)	68
4.16	Definition (Complete biorientation)	68
4.17	Definition (Strongly connected)	68
4.18	Definition (Strong component)	69
4.19	Definition (Connected)	69
4.20	Definition (Connected for digraphs)	69
4.21	Definition (Bridge)	69
4.22	Definition (Expands relation)	75
4.23	Definition (Requires relation)	75
4.24	Definition (Property graph)	79
4.25	Definition (Property graph schema)	82
4.26	Definition (Consistency of a property graph schema)	84
4.27	Definition (Intersector node)	98
Chapter 5		
5.1	Definition (Density)	107
5.2	Definition (Degree Centrality)	108
5.3	Definition (In- and Out-Degree Centrality)	108
5.4	Definition (Betweenness Centrality)	109
5.5	Definition (PageRank)	110
Chapter 6		
6.1	Definition (Learning paths)	169
6.2	Definition (Prerequisites graph)	169
Chapter 8		
8.1	Definition (Inverse Document Frequency (idf))	202
8.2	Definition (tf-idf)	202
8.3	Definition (Jaccard coefficient)	206
8.4	Definition (Cosine similarity)	207
8.5	Definition (Precision)	220

8.6	Definition (Recall)	221
8.7	Definition (Binomial coefficient)	236

