Demonstrating GraphIT – a Platform for Working with Dependency Graphs of Learning Topics

Raphael Wimmer [1], Leonie Schrod [1], and Alexander Weichart [1]

Abstract: Students' prior knowledge and goals are getting more and more heterogeneous. Thus, one-size-fits-all, same-procedure-as-every-year degree programs and courses fit the students' biographies less and less. There is little tool support for designing cohesive but flexible degree programs and courses. GraphIT is a new prototypical platform for defining curricula and courses in a dependency graph, linking learning topics to their prerequisites. The dependency graph can then be used to highlight central topics, paths between topics, and clusters of related topics. This generic approach supports educators in collaboratively modeling degree programs, lecturers in creating courses, and students in identifying personal learning paths and goals. GraphIT is built on the WikiBase platform and can be queried using SPARQL. This allows for extending and customizing schema, content, and applications.

Keywords: graphs, dependency graphs, curricula, software, WikiBase

1 Dependency Graphs for Learning Topics

University students' backgrounds and goals are becoming more and more heterogeneous [Ju23]. This requires educators to design courses and curricula that are more flexible and can be combined and re-configured depending on changing requirements. However, there is little tool support for designing such courses or degree programs. To address this problem, researchers have designed different ontologies for structuring the knowledge that students should acquire [SPJ20]. Dependency graphs (Figure 1) are a special kind of ontologies that describe which topics build upon which prerequisites. In this paper, we outline how dependency graphs of learning topics can help in planning and organizing courses and curricula and describe our implementation.

By itself, such a graph can already be used to identify central topics that should be taught early on, gaps or overlaps between courses, and learning paths that lead to a certain goal. Extending it with additional nodes and edges allows it to be used as a tool for course organization, self-directed learning, or other uses.

Over the past decade, various prototypes of such dependency graphs have been built. Lightfoot [Li14] created a dependency graph for the curricula of a college of business administration. He applied various simple graph measures, such as Eigenvector centrality, to identify importance and interconnectedness between ~70 courses. In the STOPS platform

[1] Universität Regensburg, Germany, raphael.wimmer@ur.de, https://orcid.org/0000-0001-5162-5113; leonie.schrod@stud.uni-regensburg.de, https://orcid.org/0009-0006-1762-716X; alexanderweichart@fastmail.com, https://orcid.org/0009-0005-8257-1222

doi:10.18420/delfi2025_60

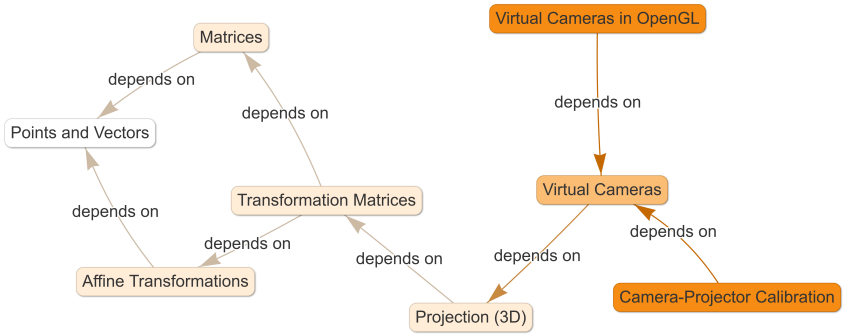

Fig. 1: A dependency graph of a selection of topics taught in a computer graphics course. Edges between nodes indicate that a topic builds on another topic. Topics with many incoming edges represent important foundations that should be learned early on.

by Auvinen et al. [APH14], nodes represent learning outcomes instead of courses. The authors mention that manually constructing such a graph is very time-consuming and that neither involved teachers nor students used the tool much. COMPBASE, an abandoned PhD project, defined an ontology of connected competences to support *portfolio learning* [DK19]. Pasterk et al. have published multiple papers on applications of graphs for modeling curricula [Ch20; PB17b]. They generate graphs from reference curricula. These graphs have two types of edges between nodes: "depends" and "expands". In the Digifit4all project[2], Pasterk et al. [Pa22] aim to build a comprehensive platform for building graphs of learning topics and integrating these in learning management systems. The platform seems to be in an unmaintained state, however. Learney, a now defunct platform[3] offered an interactive dependency graph of machine-learning foundations. For each topic, links to resources were provided, and users could mark whether they had understood a topic or wanted to learn it. As noted by multiple authors [APH14; Ch18; Ku18; PB17a], defining and building such dependency graphs is both non-trivial and tedious. Also, practical adoption of such tools has been rare [APH14; DK19]. Unlike previous approaches, we iteratively defined GraphIT's ontology while using it in teaching. We also preferred pragmatic, fuzzy definitions over formal consistency. This approach resulted in a platform that not only helps identify learning paths but also supports course planning, course organization, and learning analytics.

2 GraphIT

In GraphIT[4], nodes do not represent competences, learning outcomes or whole courses but small learning topics. As a rule of thumb, a topic typically spans about as much content as can be *explained* in a 10-20 minute talk (e. g., "Principles of Hamming Codes").

[2] https://www.digifit4all.at/
[3] https://web.archive.org/web/20240315072433/https://learney.me/
[4] https://graphit.ur.de

This does not necessarily mean that a topic can be learned in this amount of time. If we do not yet know how to break a larger topic into smaller ones, or if we don't currently see a need for that, we instead add it as a placeholder (e. g., "Mathematical Foundations"). Typically, dependencies are the central relation between nodes. However, we also use several other types of edges, such as "related to" for similar topics or "includes" for topic areas. GraphIT is built on the open-source WikiBase graph database/portal, which we extend in multiple ways. Currently, GraphIT contains three complete courses with about 400 topics. Nodes not only represent learning topics but also other concepts, such as courses, students and instructors, and resources. To interact with the graph, students can register a user account and connect "their" node to topics in the graph via "interested in" or "has completed" (Figure 2).

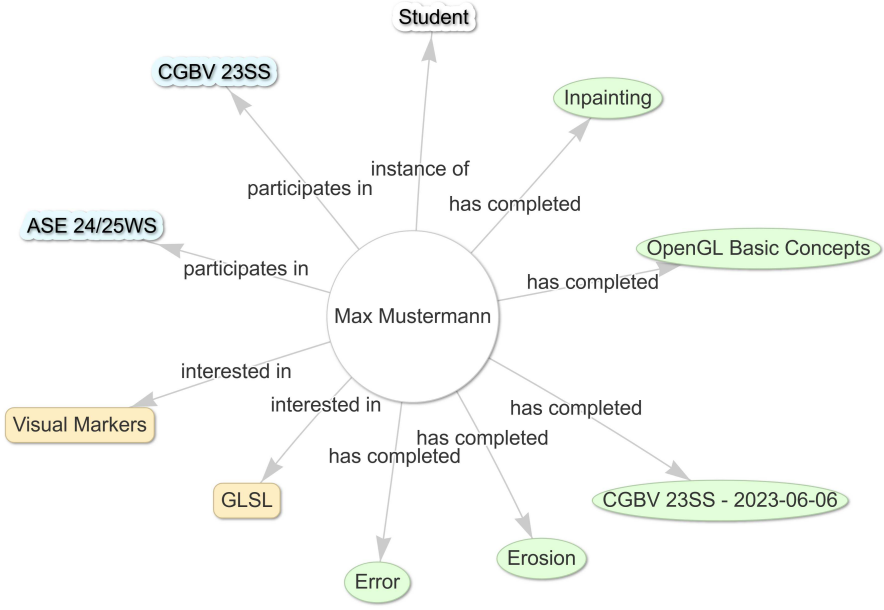

Fig. 2: On GraphIT, students can e. g., indicate topics they want to learn or have already completed, or subscribe to courses. Everything is stored as edges between the student's node and other nodes.

While we initially evaluated many graph databases, we decided to use WikiBase[5], the open-source platform underlying the WikiData[6] knowledge graph. WikiBase extends MediaWiki with a triple store, a query frontend, and various tools. While we deploy it via a customized Docker setup, there are also options for hosting Wikibase instances in the cloud. One major advantage of using WikiBase is that it offers a Wiki, user management, a query language (SPARQL[7]), simple visualizations, and an ecosystem of tools and communities out of the

[5] https://wikiba.se/
[6] https://www.wikidata.org/
[7] https://www.w3.org/TR/sparql11-query/

box. WikiBase requires no pre-defined schemas or class hierarchy. This allows us to iterate quickly and try out new ideas with minimal effort. In WikiBase, nodes are called *Items*, edges are called *Properties*. Each Item has its own wiki page that contains *Statements*, pairs of Properties and Items (e. g., "depends on": "Item Q123"). WikiBase has built-in support for qualifiers, i. e., annotations attached to edges. We use this feature for adding relevant metadata to edges. For example, for each dependency we can add a qualifier "added by: X" or "importance: essential". While WikiBase offers support for formal constraints[8], we do not use them as we are still trying out different ways of modeling relationships. Building on WikiBase allows us to easily link to content on WikiMedia Commons and to concepts in WikiData. Also, we can quickly create course pages that include auto-generated visualizations and reports (Figure 3 center/right). For example, students in a course on scientific methods were asked to mark topics from the course syllabus which they were interested in or which they did already know. With a short SPARQL query, we generated a visualization of students' interests (see Figure 3 right) and discussed it with them.

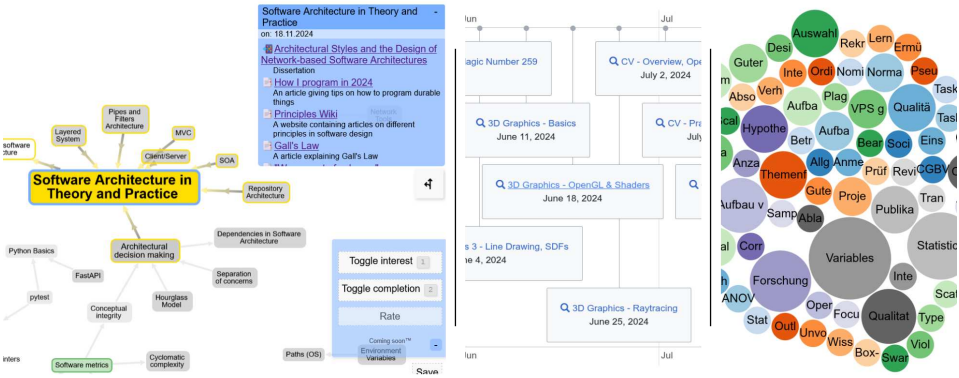

Fig. 3: We have developed custom frontends that e. g., allow users to plan their personal learning path and access accompanying materials for each topic (left). WikiBase also offers built-in visualizations for SPARQL queries which we use for displaying a course outline (center) or a list of participants' favorite topics (right).

3 Initial Experience and Outlook

We have used GraphIT for planning courses on computer graphics, software engineering, and scientific methods. Preliminary feedback from surveys and discussions indicates that students and other educators like the concept very much. However, many struggle with the amount of information represented in the usual graph visualizations. Therefore, we are currently developing custom frontends and visualizations for students and instructors (Figure 3 left). Formal evaluations will follow. All software and data are available under open licenses as outlined on the website.

8 https://www.wikidata.org/wiki/Help:Property_constraints_portal

References

[APH14] Auvinen, T.; Paavola, J.; Hartikainen, J.: 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, Association for Computing Machinery, New York, NY, USA, pp. 25–34, 2014, DOI: 10.1145/2674683.2674689.

[Ch18] Chen, P. et al.: KnowEdu: A System to Construct Knowledge Graph for Education. IEEE Access 6, pp. 31553–31563, 2018, DOI: 10.1109/ACCESS.2018.2839607.

[Ch20] Chystopolova, Y. et al.: Identification of Dependencies Between Learning Outcomes in Computing Science Curricula for Primary and Secondary Education – On the Way to Personalized Learning Paths. In (Kori, K.; Laanpere, M., eds.): Informatics in Schools. Engaging Learners in Computational Thinking. Lecture Notes in Computer Science, Springer International Publishing, Cham, pp. 185–196, 2020, DOI: 10.1007/978-3-030-63212-0_15.

[DK19] Dehne, J.; Kiy, A.: Using an Ontology Based Competence Database for Curriculum Alignment of Portfolio Based Learning. In. Proceedings of DELFI Workshops 2019. Gesellschaft für Informatik e.V.z, p. 121, 2019, https://dl.gi.de/handle/20.500.12116/27953.

[Ju23] Judel, S. et al.: Supporting Individualized Study Paths Using an Interactive Study Planning Tool. In. 21. Fachtagung Bildungstechnologien (DELFI). Gesellschaft für Informatik e.V., pp. 225–230, 2023, https://dl.gi.de/handle/20.500.12116/42196.

[Ku18] Kubekov, B. et al.: Methodology of Formation of Educational Resources On the Basis of Ontology. In: 2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT). 2018 IEEE 12th International Conference on Application of Information and Communication Technologies (AICT). Pp. 1–6, 2018, DOI: 10.1109/ICAICT.2018.8747069.

[Li14] Lightfoot, J.: A Graph-Theoretic Approach to Improved Curriculum Structure and Assessment Placement. Communications of the IIMA 10 (2), 2014, DOI: 10.58729/1941-6687.1136.

[Pa22] Pasterk, S. et al.: DigiFit4All – Conceptualisation of a Platform to Generate Personalised Open Online Courses (POOCs). In (Passey, D. et al., eds.): Digital Transformation of Education and Learning - Past, Present and Future. IFIP Advances in Information and Communication Technology, Springer International Publishing, Cham, pp. 247–258, 2022, DOI: 10.1007/978-3-030-97986-7_21.

[PB17a] Pasterk, S.; Bollin, A.: A Graph-based Approach to Analyze and Compare Computer Science Curricula for Primary and Lower Secondary Education. In: Proceedings of the 2017 ACM Conference on Innovation and Technology in Computer Science Education. ITiCSE '17, Association for Computing Machinery, New York, NY, USA, p. 365, 2017, DOI: 10.1145/3059009.3072985.

[PB17b] Pasterk, S.; Bollin, A.: Graph-Based Analysis of Computer Science Curricula for Primary Education. In: 2017 IEEE Frontiers in Education Conference (FIE). 2017 IEEE Frontiers in Education Conference (FIE). Pp. 1–9, 2017, DOI: 10.1109/FIE.2017.8190610.

[SPJ20] Stancin, K.; Poscic, P.; Jaksic, D.: Ontologies in Education – State of the Art. Education and Information Technologies 25 (6), pp. 5301–5320, 2020, DOI: 10.1007/s10639-020-10226-z.