Promoting Deep Learning Through a Concept Map-Building Collaborative Activity in an Introductory Programming Course

João Paulo Barros
Polytechnic Institute of Beja, Portugal
Center of Technology and Systems (UNINOVA-CTS) and Associated Lab of Intelligent Systems (LASI), Caparica, Portugal

Abstract

Programming courses focus heavily on problem-solving and coding practice. However, students also face numerous interrelated concepts that should be given more attention to foster more effective and comprehensive learning. Often, students only get an incomplete knowledge of those concepts and their relations as no adequate reflection is promoted or even seen as necessary. The result is a superficial surface learning about essential programming concepts and their relations. This experience report presents a learning activity to promote deep learning of concepts and their relations. The activity challenges students to specify relations between concepts. Students search definitions for a given set of concepts and define relations between those concepts in textual form. To that end, they use a freely available tool that produces a graph from textual descriptions. This tool dramatically simplifies and speeds up the creation of readable graphical representations. Although many different courses can take advantage of the presented activity, we present the activity's application to an introductory object-oriented programming course. We also present and discuss the student's feedback, which was highly positive. In the end, we provide recommendations, including possible variations. These can help educators to effectively foster active learning of concepts and their relations in their classrooms.

2012 ACM Subject Classification Social and professional topics → Computing education

Keywords and phrases active-learning, ontologies, concepts, concept maps, learning activity, object-oriented programming, oop, pedagogy, education

Digital Object Identifier 10.4230/OASIcs.ICPEC.2024.7

Funding *João Paulo Barros*: This research was funded by Portuguese Agency FCT – Fundação para a Ciência e a Tecnologia no âmbito da Unidade de Investigação CTS – Centro de Tecnologia e Sistemas/UNINOVA/FCT/NOVA, com a referência UIDB/00066/2020.

1 Introduction

Programming courses emphasize practical problem-solving projects and coding exercises. At the same time, students are exposed to and put to practical use a wide variety of interrelated concepts. However, these concepts often need additional and adequate emphasis to counter the risk of a shallow understanding of them and their mutual relations. One way is to promote active reflection and discussion about those concepts and their connection in class. This experience report presents a learning activity that fosters conceptual thinking to counter surface learning and promote deep learning about concepts identified as necessary for the course's intended learning outcomes (e.g., [12]). To that end, the teacher identifies a set of concepts and challenges the students to create their concept maps by establishing significant relations between those concepts in a (graphical) concept map. Here, the concept map is seen as a simplified ontology, as it also forces students to investigate and learn the meaning of the concepts to describe their mutual relations, thus achieving a simultaneously specific and holistic understanding of them. However, as formal descriptions of concepts and their mutual relations, ontologies are often exhaustive, hence quite complex, and around a specific area. Therefore, the respective reference tools are also complex (e.g., [21]) and do not provide a quick and straightforward way to build even simple ontologies. Here, we propose a learning activity using simple textual language to describe relations between concepts, from which a graphical representation is automatically generated. This is very significant for the activity's effectiveness, as we have observed that this fact generates a subtle but effective wow effect that promotes engagement and motivation around a topic that can easily be seen as uninteresting, unimportant, or both.

The tool that allows the graphical representation is freely and readily available on the web with no need to install software locally: it can be used on any machine where a browser is available, including mobile devices [6, 4, 5]. The activity was applied in an introductory course on object-oriented programming, a subject matter where students struggle with numerous complex and strongly interconnected concepts. Although the tool's nature could probably make it more attractive to computer science students, its ease of use suggests that it can be suitable for any topic, even outside computer science.

In the following section, we give the background and motivation for the key concepts, namely ontologies, concept maps, and their relation to the reported activity. Section 3 presents related work, and Section 4 the context where the activity was applied. Section 5 details how the activity was conducted and includes some lessons from our experience applying similar activities. Section 6 presents and discusses the observed outcomes based on in-class observation and a student survey. Finally, Section 7 provides several recommendations allowing possible activity variants and concludes.

2 Background and Motivation

An ontology is a concept with its origins in Philosophy, where it was "coined in 1613, independently, by two philosophers, Rudolf Göckel (Goclenius) in his Lexicon philosophicum and Jacob Lorhard (Lorhardus) in his Theatrum philosophicum" [23]. It seeks a definitive and exhaustive classification of entities, including the types of relations that tie them.

From an engineering perspective, ontologies are design artifacts used to describe knowledge and support knowledge-sharing activities. In this sense, their use in several areas became a trend, as testified by the many articles citing seminal work (e.g., [7] with more than thirteen thousand citations). Ontologies gained such widespread popularity that they are now defined in an international standard for vocabulary in systems and software engineering: "ontology. (1) logical structure of the terms used to describe a domain of knowledge, including both the definitions of the applicable terms and their relationships (ISO/IEC/IEEE 24765i:2020)" [11].

The above definition includes the need to define the applicable terms. Hence, in brief, an ontology becomes a set of defined terms and their (defined) relationships.

The term "ontology" is not as common in the education literature as "concept maps," "concept mapping," or even "knowledge map". In 2006, Nesbit and Adosope [18] estimated that "(...) more than 500 peer-reviewed articles, most published since 1997, have made substantial reference to the educational application of concept or knowledge maps." This may be related to the genesis of "concept mapping," as it has emerged "inside" education research, namely to study changes in students understanding of science concepts along twelve years of schooling [19]. However, concept maps, although not so formally defined and discussed, can be seen as a simple way to create ontologies where the "concepts" are the "terms" and the "map" establishes the "relationships" among "concepts." In our reported activity, the used "concept maps" are more straightforward and shorter because the definitions for the "concepts" are not mandatory. However, students must still know enough about each concept to create meaningful relations among them.

More significant for our work is the fact that "conceptual mapping" is an effective learning activity: the already cited meta-analysis by Nesbit and Adosope [18] concluded that "(...) in comparison with activities such as reading text passages, attending lectures, and participating in class discussions, concept mapping activities are more effective for attaining knowledge retention and transfer."

By promoting deep learning of concepts, our reported activity can easily allow for the identification of students' misconceptions, a common topic in the literature (e.g., [14],[13],[22]) and provide a basis for a new form of "misconception-driven-feedback" [8]. In [14], formal interviews were used to identify misconceptions; in [13], the objective was to validate a methodology for building a concept inventory dedicated to OOP; in [22], multiple-choice-based questions were used to identify misconceptions. The activity we report in this paper allows a quick, in-class, collaborative identification of misconceptions in any domain or subdomain with the added anecdotal benefit of directly promoting learning and being more engaging than a traditional assessment element such as a multiple-choice or oral exam. Additionally, the reported activity also provides a tool for assessing and identifying incomplete and inconsistent student mental models, another significant problem (e.g., [17]).

While promoting critical and conceptual thinking, the created concept maps also offer an index and a structured map of the concepts students have to know and apply. Hence, they provide a "structural view" (concepts and relations) that complements the "behavioural view" (coding), where students spend most of their time.

Finally, students creating relations between concepts can also be seen as a direct application of constructivism, a theory of learning that claims that students construct knowledge rather than merely receive and store knowledge transmitted by the teacher[2].

3 Related Work

In a literature survey on the use of ontologies in education, the authors identify numerous uses, but primarily for describing learning domains [24]. Surprisingly, the authors go to the point of asserting that "While developing ontologies, one must use a programming language." Thus, they somehow impose the use of computer support for creating ontologies. They identify several types of ontologies and the methodologies used to define them. The authors also present an overview of existing systems that use ontologies in the education domain. In that context, it is easy to find papers proposing ontologies for one or more domains in the education area. In the area of programming, one focus is the languages themselves (e.g., [15]).

Also, there are numerous cases of using concept mapping and concept maps as part of learning activities (e.g., [1]). However, it is important to stress that *concept map* is a term nowadays usually associated with the work by Novak in the 70s (e.g., [25]) for a specific method to represent knowledge by relating concepts. More specifically, "concepts are arranged hierarchically with the most general, most inclusive concept at the top, and the most specific, least general concepts toward the bottom." (in [20]). Novak and Cañas claim that "Concept mapping was invented in 1972". However, it seems clear that concept mapping can be seen as part of what an ontology is, as presented in the previous section: "classification of entities including also the types of relations that tie those ."In fact, we also used the name "concept map" to better explain to students what they were going to create: they map concepts to other concepts to build a simple ontology. We also alert them that, in the literature, the name "concept map" is frequently used for a specific way to map concepts.

Unlike previous works, our approach is based on a freely available tool that automatically generates graph layouts from textual descriptions. This automation streamlines the creation of concept maps, increasing student engagement and motivation. Additionally, automatic graph layout generation from text is a compelling topic in itself, particularly for computer science students, some of whom have expressed interest in applying the tool across courses. We provide detailed instructions and variations for the activity, enabling teachers to adapt it to various contexts.

4 Context

The reported activity was applied in 2022/2023 in an introduction to object-oriented programming course. The course is offered in the second semester of a three-year computer science bachelor's degree program at a small university with around three thousand students in total, of which around two hundred are in the computer science program. It is a second course on programming as the students already have a first course in the first semester. However, students may have failed the first course and still try to do the second, as allowed. The present article reports the activity conducted with the students in the 2022/2023 course edition. This activity already integrated some lessons learned in previous applications of similar activities in 2021/2022. These lessons are presented in Subsection 5.2.

5 Concept map creation activity

This section describes the activity as applied in the 2022/2023 course edition. As already stated, this incorporates lessons learned in previous applications in 2021/2022, which were briefly reported elsewhere [*anonymized*]. This first subsection should provide enough information to allow educators to reproduce the activity in their courses. The second subsection presents lessons learned from the 2021/2022 edition that motivated some changes in the 2022/2023 edition.

5.1 Application

The reported activity was designed and applied as an in-class face-to-face activity. Nevertheless, with minimal modifications, it can surely be applied in synchronous remote classes and, with minor changes, even as an asynchronous remote activity. The activity steps were the following:

1. The motivation for the activity is explained to the students, namely the importance of learning the concepts and their relations;
2. The GraphViz online tool [6, 4, 5] is briefly presented by explaining the given base description and how it quickly and automatically generates the respective graph; the set of concepts related to previously studied contents are presented as nodes in the graph; the syntax to define relations between concepts is also exemplified;
3. Each student is given 20 minutes to create a graph relating concepts that, for the sake of simplicity, we call a "concept map";
4. Each student gets together with another colleague to create an improved concept map; another 20 minutes are given;
5. Each pair of students gets together with another pair to create an even better concept map; each group is given an additional 20 minutes to create an improved concept map;
6. Each group of four publishes, in a shared document, the resulting concept map for all to see;
7. The teacher presents and discusses with the class one or more of the shared concept maps.

In step (1), the teacher notes that students have already used and applied numerous concepts that should be better understood. To that end, they must search for and relate the definitions, thus creating their own concept map. In this way, they should better understand those concepts and their mutual relations. Regarding step (2), the teacher presents a slide with the link and the respective QR code to the start graph in the online tool (see Fig. 1). The graph layout automatically generated is presented in the center of the slide. A giant QR code (omitted in the figure) is presented to facilitate students accessing the start graph in the online web tool. The concepts to be related are already presented in the context of the tool to be used, more specifically in the listing on the left side of the slide in Fig. 1. For readability purposes, the listing is also presented after the figure. In this example, the specified graph is directed (`digraph`) (arcs are "one side arrows"). Nodes are shaped like ellipses. Those nodes are the concepts the students should relate: `value`, `variable`, `constant`, ..., `operator`. Then, four examples of arcs are also presented. As the syntax is straightforward, for each new relation, students only have to copy and paste one of the lines 22 to 25 and change the node names and the parameter `label` to assign the intended meaning to the relation.

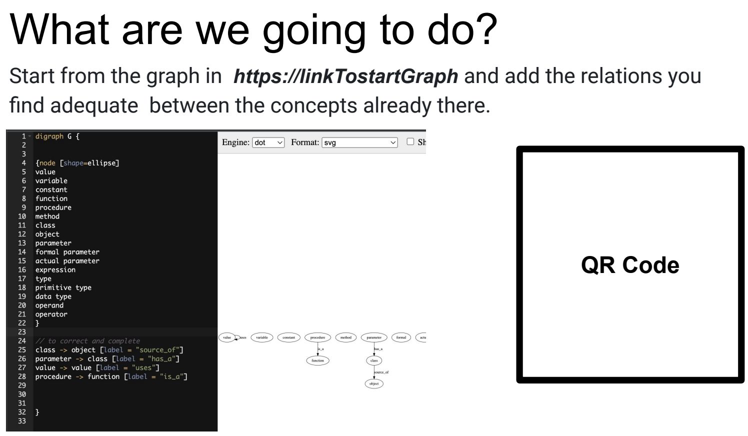


Figure 1 Introductory slide.

```
1  digraph G {
2  { node [shape=ellipse]
3  value
4  variable
5  constant
6  function
7  procedure
8  method
9  class
10 object
11 parameter
12 formal parameter
13 actual parameter
14 expression
15 type
16 primitive type
17 data type
18 operand
19 operator
20 }
21 // to correct and complete
22 class -> object [label = "source_of"]
23 parameter -> class [label = "has_a"]
24 value -> value [label = "uses"]
25 procedure -> function [label = "is_a"]
26 }
```

Fig. 2 presents the graph layout generated from the description. It shows the concepts as graph nodes (ellipses) and the four relations as directed arcs.

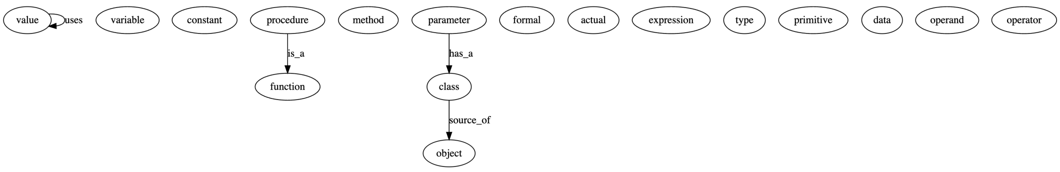


Figure 2 Start graph layout.

The concepts are nodes in the *dot* description language, and, in the end, a set of four relations is presented. These are guaranteed to be syntactically correct but not semantically, as students are asked to correct and complete them. For example, in line 27 of the presented list, a procedure is (wrongly) defined as being a function. The teacher exemplifies how students can quickly create new relations by copying and pasting one of the relations in the last lines of the listing and changing the names. Next, the teacher follows the link and exemplifies some changes to the graph description so that students can see the immediate generation of the graph layout.

In steps (3), (4), and (5), the 1-2-4-all method [16] is applied to guide students in producing their concept maps: first individually, then in pairs, and finally in groups of four, students create graphs that relate the given concepts, thus creating their concept maps that are then shared with all. From our experience, it is essential to briefly explain the four method stages, with the help of a slide (see Fig. 3), and also that students know, all the time, the stage they are in (individual, pairs, or groups of four). For that reason, one slide with a countdown timer is used for each stage. Fig. 4 shows the slide for the first stage (the individual work one). It also includes a link to the shared document and the start graph.

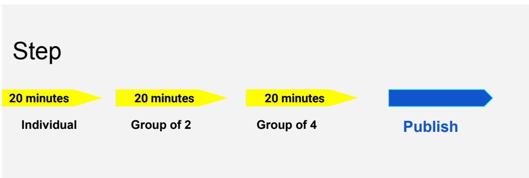


Figure 3 Slide for presenting the 1-2-4-all method to students.

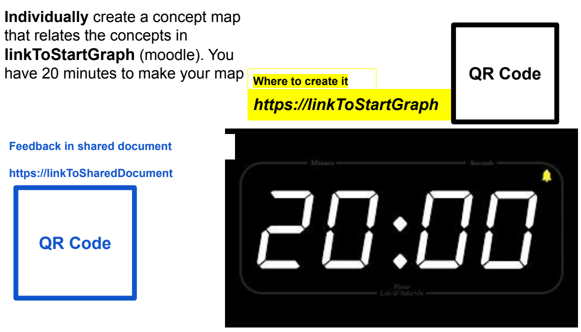


Figure 4 First stage slide.

Next, in step (6), each group shares (anonymously if they prefer) the result with the class in an online document. This document remains accessible as the result of the class activity and can be consulted any time after, in, or outside class.

Finally, in step (7), the teacher chooses one or more concept maps and discusses their contents with the class.

5.2 Lessons learned and practicalities

Preliminary versions of the presented activity were applied in the 2021/2022 academic year, together with a closely related activity: the presentation of a ready-made concept map. From those applications, some observations were made, and some lessons were learned:

- With no rule for collaboration, most students were quite individualistic, and collaboration was minimal;
- One student complaint they had too much time alone to create the concept map;
- Besides the tool's simplicity, some students asked for a brief explanation of its functioning;
- Some students asked for an exemplary concept map made by the teacher to be studied after the activity.

The activity here reported already incorporated the following strategies to answer the above points: regarding (1) and (2), students now start individually but then proceed in pairs and groups of four; regarding (3), the teacher gave a brief explanation about how to use the tool; regarding (4), the teacher provided an exemplary concept map after the activity. The following section presents the results of the reported activity as applied in the academic year 2022/2023.

6 Observed outcomes

This section presents the results as students' perceptions. These were collected in two ways: (1) in class, along the activity, students were randomly and informally asked about their opinions, and the teacher observed their performance; (2) a post-class questionnaire was applied to all 70 students in the activity and 42 students responded.

6.1 In-class observation

No significant difficulty was observed throughout the class, as no student felt impeded from proceeding. There were just a few questions about what was allowed as relations, primarily due to a lack of attention to the initial explanation and because some students wanted to create additional types of relations, e.g., using bidirectional arcs. Most students were very engaged, and one reason was the wow factor associated with observing the automatic graph layout generation after each change in the textual description.

Students vocalized a few opinions along the activity execution that we list next:

- For learning, it is also useful to compare the created concept maps with other concept maps at the end of the activity; only the creation would be insufficient;
- To see an exemplary solution would be very useful;
- One previous solution is not so useful, but it can be helpful to better understand what the objective is.

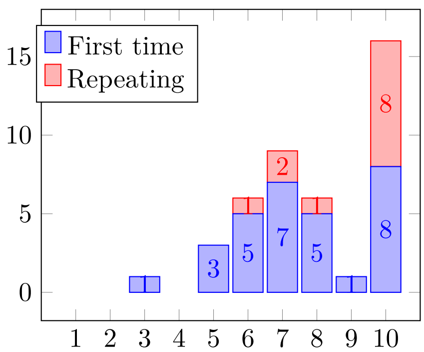
As students were given the possibility to create new relations (new labels in arcs), most did it. At the end of the class, the teacher's perception was that all students had found the activity very positive and helpful.

6.2 Student Survey

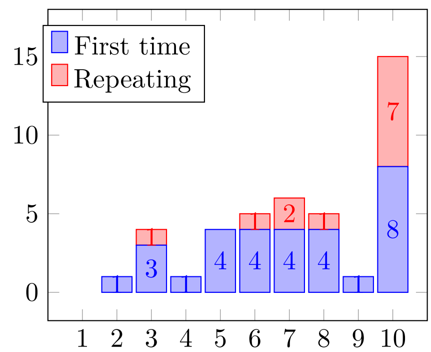
After the classes, all 70 participating students were invited to fill out a short questionnaire. We got 42 answers: 30 were first-time students, and 12 were repeating students. In an online questionnaire, they were asked to grade the following four assertions on a scale from 1 (totally disagree) to 10 (totally agree):

1. "It is useful to see and discuss a concept map in class.";
2. "It is helpful to have concept maps for studying outside class.";
3. "Concept maps with other sets of programming concepts can help understand them better.";
4. "The concept map was asked to be done in three sequential steps: individually, in pairs, and groups of four. That method is something to repeat.".

The survey results are presented as stacked bar charts in figures 5a, 5b, 6a, and 6b.

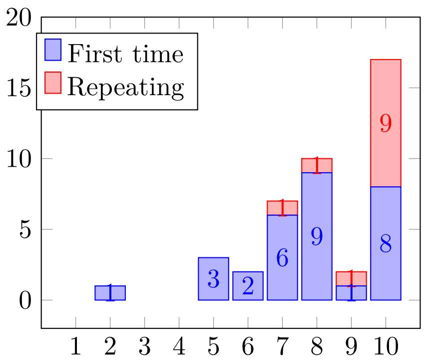


(a) Results for the assertion "It is useful to see and discuss a concept map in class." 1-totally disagree to 10-totally agree.

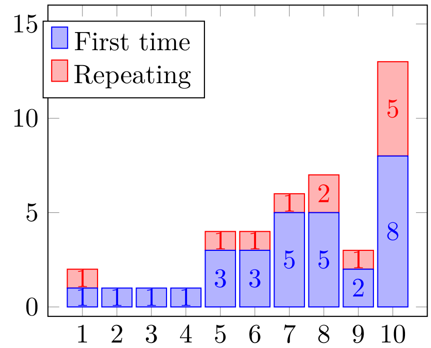


(b) Results for the assertion "It is helpful to have concept maps for studying outside class." 1-totally disagree to 10-totally agree.

Figure 5 Grading usefulness and helpfulness.



(a) Results for the assertion "Concept maps with other sets of programming concepts can help understand them better." 1-totally disagree to 10-totally agree.



(b) Results for the assertion "The concept map was asked to be done in three sequential steps: individually, in pairs, and in groups of four. That method is something to repeat." 1-totally disagree to 10-totally agree.

Figure 6 Other concepts and process.

In all four, the most frequent answer was "totally agree." Also, there is a clear agreement with all four assertions. Students' opinions varied more regarding the usefulness outside class but still had a very high level of agreement: only 10 in 42 graded it between 1 and 5.

Regarding the fourth assertion, about the applied method, two students answered "totally disagree": as made clear in their open answers, the first-time student stated a preference for an unrestricted number of students in the groups; the repeating student mentioned a timetable overlap with another class, a clearly unrelated motif.

In the same survey, students were also given the (optional) opportunity to leave some free comments about the activity. Next, we list the comments:

- "In my opinion, all the results obtained should be discussed to take the best possible advantage.";
- "I think it's a very good method, it helps a lot those who do not feel very comfortable in the area to become more interested.";
- "Do more activities in class to encourage students to interact with each other.";
- "I think we should have more similar classes, we really lack communication in the classroom, and group classes bring a different dynamic to the classes and to our own way of learning.";
- "It would be important for the teacher to show a map made by himself in order to be able to compare the maps developed by the students.";
- "It's great. We should do it more often.".

The main suggestion by students was for the teacher to make available an exemplary solution so that they could compare it with their own. This is probably due, at least in part, to the fact that this is not an exercise that they can "run" and check if it is correct, as they are used to when programming.

Next, we present some main recommendations, most of which are optional additional steps in the activity application.

7 Limitations

The current study has several limitations that should be acknowledged. First, the sample size was relatively small, with only 30 first-time and 12 repeating students participating in the study. These numbers may limit the generalization of the results, as the sample may not be representative of the broader student population. Additionally, a long-term follow-up assessment is needed to determine the sustained impact of the proposed activity on students' learning outcomes. Future research should replicate this study with a more extensive and diverse sample and incorporate long-term follow-up assessments to evaluate the enduring effects of the activity.

8 Recommendations and Conclusions

The reported activity was applied in two 100-minute classes. However, with more class time or reducing the time for each stage, several additional sub-activities can be conducted to boost the activity objective. Here, we list the ones we have identified as potentially interesting. Some result from additional reflection after applying the reported activity and listening to students' opinions. Others were identified initially but left out due to timing constraints. In any case, the addition of one or more of the presented sub-activities will probably promote the activity objective. Time seems to be the only significant restriction to their application. Next, we present the identified sub-activities grouped in three contexts: (1) before the class where the activity will take place; (2) during the activity itself; (3) after the presented in-class activity, but still in class; (4) as autonomous work, after the class where the activity took place.

1. **Before the class where the activity will take place**
   - **Videos about concepts.** Make available one or more videos about the concepts and reward comments students make to those videos or to the colleague's comments. A platform like VideoAnt [9, 3], or a simple online forum for this specific task can support this ;
   - **Videos about GraphViz online tool.** If more sophisticated relations are going to be asked, then one or more tutorial videos about the GraphViz online tool and the dot language may also be made available before the in-class activity;
   - **Initial quiz.** Answering some questions about the concepts before the activity can be useful for the teacher as a diagnostic assessment and for the students as a formative assessment;
   - **Individual stage before class.** The individually created concept map can be done as a pre-class exercise, leaving class time for group work and thus increasing collaboration time;
   - **Exemplary concept maps.** To better clarify the intended activity object and objective, one or more exemplary concept maps relating different but well-known sets of concepts can be given in advance;
2. **During the activity itself**
   - **Add concepts' definitions.** Strictly speaking, each ontology should include the concepts' definitions; here, we assumed students searched those definitions so that they could relate them, but we did not mandate them to include those definitions; however, this can be easily achieved by adding, in the dot description, links to webpages containing the student's definition of each concept;
   - **Closed set of relations.** Instead of letting students create new relation types, students may be asked only to use the given set of relations types; this has the advantage of allowing an easier measuring of outcomes, which can also simplify comparability and grading; in fact, the use of distinct and closed sets of relations and concepts can provide a useful formative assessment tool based on the observation and recording of students doubts, difficulties, and achievements;
3. **After the presented in-class activity, but still in class**
   - **Public presentation.** After the groups publish their ontology in the shared space, a speaker from each group can be asked to present the group's work;
   - **Comments on the presentations** After each presentation, all the students and the teacher can provide comments and discuss the merits and pitfalls of the presented ontology, thus providing an additional opportunity for clarification and discussion;
   - **Improvement after presentations.** The groups can be allowed to improve their concept maps based on the received comments from all students and the teacher; the comments can be oral or written, and their quality graded by the teacher or students as a reward and incentive;
   - **End of activity quiz.** Especially if an initial quiz was done, an end-of-activity quiz can be useful for the teacher and for the students as a formative assessment to answer some questions about the concepts and compare the result with those in the initial quiz;
   - **Discussion about an exemplary concept map** The teacher presents and discusses a concept map of his own with the students.
4. **As autonomous work, after the class where the activity took place**
   - **Commenting a concept map.** After the synchronous class, the teacher makes available a concept map of his/her authorship for the trainees to comment on; this can be supported by a specific tool (e.g., [10]) or by an online forum.

In the reported experience, the presented activity was perceived as valuable by the students and the teacher as a way to actively foster the deep learning of concepts and their relations in an introductory object-oriented programming course. Using a free online tool to automatically generate graph layouts significantly accelerated and simplified the creation of the graphical representation and promoted engagement. Its simplicity should allow the use of the activity in many other courses, including outside computer science.

References

1 Olusola O. Adesope and John C. Nesbit. *A Systematic Review of Research on Collaborative Learning with Concept Maps*, pages 238–255. Handbook of Research on Collaborative Learning Using Concept Mapping. IGI Global, Hershey, PA, USA, 2010. doi:10.4018/978-1-59904-992-2.ch012.

2 Mordechai Ben-Ari. Constructivism in computer science education. In *Proceedings of the Twenty-Ninth SIGCSE Technical Symposium on Computer Science Education*, SIGCSE '98, pages 257–261, New York, NY, USA, 1998. Association for Computing Machinery. doi:10.1145/273133.274308.

3 Digital Education and Innovation, College of Education and Human Development, University of Minnesota. Videoant, 2023. Available at https://ant.umn.edu/, accessed on 2023/08/15.

4 John Ellson, Emden Gansner, Lefteris Koutsofios, Stephen C. North, and Gordon Woodhull. Graphviz— open source graph drawing tools. In Petra Mutzel, Michael Jünger, and Sebastian Leipert, editors, *Graph Drawing*, pages 483–484, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

5 Emden R. Gansner and Stephen C. North. An open graph visualization system and its applications to software engineering. *Software: Practice and Experience*, 30(11):1203–1233, 2000. doi:10.1002/1097-024X(200009)30:11<1203::AID-SPE338>3.0.CO;2-N.

6 Graphviz online, 2022. Available at https://dreampuf.github.io/GraphvizOnline/, accessed on 2023/08/17.

7 Thomas R. Gruber. Toward principles for the design of ontologies used for knowledge sharing? *International Journal of Human-Computer Studies*, 43(5):907–928, 1995. doi:10.1006/ijhc.1995.1081.

8 Luke Gusukuma, Austin Cory Bart, Dennis Kafura, and Jeremy Ernst. Misconception-driven feedback: Results from an experimental study. In *Proceedings of the 2018 ACM Conference on International Computing Education Research*, ICER '18, pages 160–168, New York, NY, USA, 2018. Association for Computing Machinery. doi:10.1145/3230977.3231002.

9 Bradford Hosack. Videoant: Extending online video annotation beyond content delivery. *TechTrends*, 54(3):45–49, May 2010. doi:10.1007/s11528-010-0402-7.

10 Hypothesis, Inc. Hypothesis, 2023. Available at https://web.hypothes.is/, accessed on 2023/08/15.

11 ISO/IEC/IEEE 2017. Systems and software engineering – Vocabulary. Standard ISO/IEC/IEEE 24765, International Organization for Standardization, Geneva, CH, 2017. URL: https://standards.iso.org/ittf/PubliclyAvailableStandards/c071952_ISO_IEC_IEEE_24765_2017.zip.

12 Gregor Kennedy John Biggs, Catherine Tang. *Teaching for Quality Learning at University*. McGraw Hill, 5 edition, 2022.

13 Henry Julie, Dumas Bruno, Heymans Patrick, and Leclercq Tony. Object-oriented programming: Diagnosis understanding by identifying and describing novice perceptions. In *2020 IEEE Frontiers in Education Conference (FIE)*, pages 1–5, 2020. doi:10.1109/FIE44824.2020.9273990.

14 Lisa C. Kaczmarczyk, Elizabeth R. Petrick, J. Philip East, and Geoffrey L. Herman. Identifying student misconceptions of programming. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education*, SIGCSE '10, pages 107–111, New York, NY, USA, 2010. Association for Computing Machinery. doi:10.1145/1734263.1734299.

15 Ming-Che Lee, Ding Yen Ye, and Tzone I Wang. Java learning object ontology. In *Fifth IEEE International Conference on Advanced Learning Technologies (ICALT'05)*, pages 538–542, 2005. doi:10.1109/ICALT.2005.185.

16 Henri Lipmanowicz and Keith McCandless. 1-2-4-all – liberating structures including and unleashing everyone, 2023. Available at https://www.liberatingstructures.com/1-1-2-4-all//, accessed on 2023/08/15.

17 Syeda Fatema Mazumder. Investigating the role of explanative diagrams as a representation of notional machine on a novice programmer's mental model. In *Proceedings of the 17th ACM Conference on International Computing Education Research*, ICER 2021, pages 409–410, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3446871.3469775.

18 John C. Nesbit and Olusola O. Adesope. Learning with concept and knowledge maps: A meta-analysis. *Review of Educational Research*, 76(3):413–448, 2006. doi:10.3102/00346543076003413.

19 Joseph D. Novak. Concept mapping: A useful tool for science education. *Journal of Research in Science Teaching*, 27(10):937–949, 1990. doi:10.1002/tea.3660271003.

20 Joseph D Novak and Alberto J Cañas. The origins of the concept mapping tool and the continuing evolution of the tool. *Information Visualization*, 5(3):175–184, 2006. doi:10.1057/palgrave.ivs.9500126.

21 Protégé, 2020. Available at https://protege.stanford.edu/, accessed on 2023/08/17.

22 Vijayalakshmi Ramasamy, Mourya Reddy Narasareddygari, Gursimran S. Walia, Andrew A. Allen, Debra M. Duke, James D. Kiper, and Debra Lee Davis. A multi-institutional analysis of cs1 students' common misconceptions of key programming concepts. In Maria Virvou, George A. Tsihrintzis, Nikolaos G. Bourbakis, and Lakhmi C. Jain, editors, *Handbook on Artificial Intelligence-Empowered Applied Software Engineering: VOL.2: Smart Software Applications in Cyber-Physical Systems*, pages 127–144. Springer International Publishing, Cham, 2022. doi:10.1007/978-3-031-07650-3_8.

23 Barry Smith. *Ontology*, chapter 11, pages 153–166. John Wiley & Sons, Ltd, 2004. doi:10.1002/9780470757017.ch11.

24 Kristian Stancin, Patrizia Poscic, and Danijela Jaksic. Ontologies in education – state of the art. *Education and Information Technologies*, 25(6):5301–5320, November 2020. doi:10.1007/s10639-020-10226-z.

25 Ling Xu, Ming-Wen Tong, Bin Li, Jiang Meng, and Chen-Yao Fan. Application of concept map in the study of computational thinking training. In *2019 14th International Conference on Computer Science & Education (ICCSE)*, pages 454–459, 2019. doi:10.1109/ICCSE.2019.8845505.