Graph Theory in High School Education

D. Lessner
Department of Software and Computer Science Education, Faculty of Mathematics and Physics, Charles University, Prague, Czech Republic.

Abstract. Graph theory is not being taught on Czech high schools. The reasons seem to be rather historical than factual. In this contribution, we discuss the usefulness of elementary Graph theory for High school students. Then we formulate the targets and objectives of this education. Core of this contribution is topological sorting. It is an example of an advanced topic in graph theory, which is still comprehensible for high school students. Moreover, it is potentially useful in their life also out of school. We explain the problem, its applications, and three approaches to the solution along with comments regarding introduction of such topic into high schools. Last part of this contribution explains the critical path method, another advanced topic strongly bond to the idea of representing tasks as graphs.

Introduction

Computer science is not a part of high school education[1] in Czechia [*Lessner*, 2010]. We believe it to be a mistake. Our research addresses this issue. Graph theory is one of the topics we would like to introduce to our schools. In this contribution, we provide some more details to this specific topic. It shows our point of view on graphs for the sake of high school education.

First we clarify what we consider to be the basics of graph theory. For high school students the classical $G = (V, E)$ approach is rather confusing. What they really need to know is that graphs are structures to represents relations between objects and that it is useful to try to draw them. Then we examine the situation of graph theory on high schools. Graphs are not included in the Framework Education Programme (fundamental curricular document for our high schools, [*VÚP*, 2007]) nor other programmes ([*CERMAT*, 2010]) explicitly, yet they are used in many areas intuitively. Sadly, there is no awareness of some unifying theory which would allow knowledge transfer between different areas using graphs. After explaining some more reasons why graph theory should be introduced to high school students we describe the main educational targets.

After these preparations, we describe a more advanced topic in closer detail. We have chosen topological sorting for this purpose. The problem is to find an order of vertices in oriented graph which would respect the edges orientation. The usual idea is that vertices are tasks and edges their dependencies. We will show three approaches to solve this problem and discuss the interesting points from the high school point of view. Developing the problem of topological sorting into an even more useful idea we introduce the notion critical path, as well as a way how to find it and use it. At the end we summarize the whole contribution.

Graph theory on high schools

In this section we discuss graph theory education on high school level in Czechia. At first we briefly revisit graph theory itself. Then we look into the recent situation on out high schools regarding graph theory. Then we can discuss the reasons why to teach about it as well as specific educational targets.

Here we revisit the basics about graph theory. At first we shall make clear that graph theory in computer science is not about graphs of functions or charts. Graph is an abstract structure

[1] Secondary education, i. e. 15 – 19 years old in their 10. – 13. year of study.

to represent relations between pairs of objects. This is very general and leads to extreme applicability in a wide range of areas. Among the advantages we see also visuality. Formally, graphs are sets of vertices and edges (i. e. pairs of vertices). This would be represented by some sequence of symbols. However, it is much more useful for general population to represent graphs (i. e. relations between objects) in form of a drawing. This way one can see much deeper into the structure of represented relations and find much easier what he seeks.

Among the basic notions to study are graph, vertices and edges of course, including orientation, degree and score. Higher terms are path, cycle, coloring and components. There are some basic graph types we distinguish, mainly complete graphs, trees and regular graphs. An important topic is graph representation for the sake of further algorithmic processing, e. g. in computers, where drawing is not really helpful. Among processing we are interested in various types of systematic traversing and searching. These may also lead to solve more advanced problems, like topological sorting, described later in this contribution.

Application of graphs involves nearly any area which deals with some kind of binary relations between objects. That is actually almost any area of human activity. Classic application is in maps an plans, where vertices represent places (e. g. towns) and edges their connections (e. g. roads). This is a very helpful metaphor used to think about graphs. Another applications rise from the same idea. We do not connect only places on a map, there are also pipes and tanks or electric circuitry. More applications are mentioned in the following subsection.

Graph theory in official curricula

Let us see the situation of graph theory on Czech high schools. Shortly, it is not there. This is surprising considering the usefulness of this topic and situation abroad [*Gal-Ezer et al.*, 1995]. On the other hand, it is no surprise at all, if we consider the overall situation and historical development. Last years, maths related education is going through severe reduction of both hours and topics to teach. Interested teachers and researchers focus rather on saving what is left than on introducing new topics. This shows also the historical background. Graph theory has simply never been a part of high school programmes. The field itself, regardless of how rich and important results, is much younger compared to the classical parts of high school mathematics. Further, should it be officially included, it might be of no effect at all. Without introducing CS itself, graph theory would most likely become a part of mathematics. The most important approach to the structure of mathematical programme in Czechia is respecting historical development of mathematics. Therefore graphs would come after the other topics. In practice, the last chapters are being taught usually only in specialized seminars for students who have chosen mathematics as to be a part of their school leaving exams or might need advanced mathematics for their university studies. These seminars focus mainly on the most important chapter of the end of high school level mathematics, which is differential calculus. Next to it, graph theory would be very likely left for self-study.

Although graph theory is not explicitly mentioned anywhere in the high school curriculum, we may look for its usage, even in seemingly unrelated subjects. The finding is disturbing. Graphs are being used widely throughout virtually every school subject on intuitive level. We will mention only a few blatant and obvious examples. In organic chemistry, students draw molecular structures. Simple math is used to find formulas which can not exist (given the number of atoms and their connectivity). In biology, taxonomy and genetics uses trees. Metabolic network is a graph. Royal family trees are needed in history for some nontrivial cases. Usage in geography and cartography is clear. In language, syntactic structure of a sentence is represented using a tree. Introduction to combinatorics is based on routes and intersections. Last application to mention is one of the recent hits in primary and secondary education in Czechia, mind mapping. For a computer scientist, mind map is nothing special but a tree modification. Despite of all these examples students encounter, they are unaware of any unifying theory behind. This limits their ability to transfer any knowledge or skill (e. g. tree characteristics or

systematic searching) to some different area.

Reasons for including graphs into high school education

We have described applications of graphs in high school education. Considering how frequent they are, they stand as one of the reasons why to include graph theory into high school programmes. Here we describe also other reasons for it. Graphs represent a new structure to work with, different from numbers or other classical structures from high school mathematics. Graphs seemingly require a different way of thinking. This might provide poor skilled students with a new chance to catch up in mathematics. Even though the structure differ and it may be mostly about drawing, the reasoning behind is the same as everywhere else. The same idea works with algorithms here. They seem to be somehow different, working with vertices and edges instead of numeric values. Yet, this is no fundamental difference here. Graphic algorithms give the opportunity to notice this and therefore understand better what an algorithm itself is.

It is important to remark that even though the basics are very intuitive, graph theory naturally provides advanced topics of various difficulty. Most of them have their important applications as well as theoretical impact. One of the many examples would be the notion of planarity. Many students think of graphs as of their planar drawing. This is no mistake, yet they shall know this approach has its limits. It is crucial to discover that many different drawings of one graph exist and therefore a drawing is nothing more than a visualisation. It is also important to experience that there are graphs with no correct drawing at all, yet algorithms work in the very same way. This is about planarity – a natural and useful notion, yet not trivial at all. Another such topic is isomorphism of graphs. We believe it to be at the limits of high school students capabilities.

Concluded, the reasons for teaching graph theory is in their practical usefulness as well as in their challenging theoretical background.

Education targets

Here we introduce shortly the main targets specific for graph theory education, as we have developed them so far. They are made compatible with the official high school programme [*VÚP*, 2007]. To begin with, students shall recognize the appropriate situations to apply graphs (i. e. not regarding city plans only), and they shall also apply them. This means they shall be able to read relations from given drawn graphs as well as denoting them (both by drawing and symbolically). They shall be able to use some unified terminology. This does not necessarily mean the official terms, especially in cases they are not natural in our language (as "graph" itself). The point is smooth communication when using graphs. They should be able to use drawing and sketching to their advantage, yet understand the unavoidable need to rely on reasoning in certain situations. Further processing includes translating notations (natural language description, drawing, list, matrix) and systematic traversing. This does not mean they shall know the exact way of depth first search for example. The point is to make them understand a few key points which will make their approach systematic (i. g. marking open and closed vertices) and which will enable them reinvent the algorithm if needed.

On higher level, they should be aware of some meta knowledge and heuristics, and be able to use them appropriately. They should be aware that there are many ways to achieve the same goals (regarding both graph drawing and its algorithmic processing). This goal however does not imply these ways themselves to be the same as well. They may differ significantly, mostly in difficulty and efficiency. Influenced by all the examples in graph theory, students shall know it is good to work systematically and in phases, virtually every time when it is possible.

Further, they shall understand the importance of preprocessing data and other preparations preceding work itself. One of the many examples is topological sorting, discussed below in this contribution. On the same example, they shall learn that a greedy approach may work well.

By greedy we mean fixating partial solutions and no turning back (unlike in backtracking). This usually require some more work with choosing which partial solution to pick. Usually the least injuring one is found and picked. Still, we would like them that such an argument is not necessarily correct and they shall rather seek guarantee of a partial solution correctness (and guaranteed inclusion in a correct and complete solution).

Educational targets described above combine both practical usability of graphs and theoretical knowledge. These two shall strengthen each other synergistically. Practice leads to problems, solved using theoretical results, enabling more practice, and so on. This enhances students lives, providing them with tools to solve problems (using graphs), and make them understand the importance of quality theoretical thinking at the same time.

Example: Topological sorting

We have chosen the problem of topological searching as an example for this contribution. It represents a problem which is understandable, challenging, yet well solvable and widely applicable. It is well known, therefore we will not dive into exact details. We rather save space for what is interesting in high school context. Detailed explanations can be found in [*Kučera*, 1983] and [*Töpfer*, 2007]. In this sections we introduce the problem and the main motivation for its solving, three main approaches to find the solution and a few further topics.

The problem

On input we have a directed graph. We search for an order of vertices which would respect edge orientations. More formally, for a given $G = (V, E)$, we search for a $\phi : V \to \aleph$, such that $(v_1, v_2) \in E \Rightarrow \phi(v_1) < \phi(v_2)$. The usual motivation for this problem is about tasks (represented as vertices) and their time dependencies (edges), such as getting dressed, building a villa or constructing a plane. It is not wise to put on shoes before socks, yet shoes are without any direct relation to a tie. With a complex project, it is easier to find these dependent pairs than the overall order of tasks.

Such a motivation is relevant to high school students lives. Not only do they encounter such problems occasionally, it is also important that the problem setting is comprehensible for them. Moreover, they may expect such (and more complex) problems in their future.

The solution

We will discuss three main approaches to solve the problem of topological sorting. They differ in many aspects, including efficiency. The first approach to consider is based on brute force. That would be an average student's first choice to examine. Try a few permutations of vertices and examine whether they satisfy given conditions (partial order implied by the graph). Obviously, with $n!$ possible orders, this may take very long, and moreover, the student will soon feel that he examines needless number of possibilities. He might notice some repeating patterns or even partial solutions. Trying to combine and complete them may lead to a solution, as well as to a few questions: How can we be sure it is correct? How do we do this systematically with different input data?

During their work on a specific input, they might discover parts of their solution which are not going to change anymore. Specifically, they should notice the special characteristics of the first (or last) task to be done: they have no predecessors (or successors). This may lead to find the desired order systematically instead of a sequence of trial and errors. Excluding already ordered vertices from the graph (as well as incidental edges), we may proceed recursively. The graph is smaller, the idea is the same. Task with no predecessor may be put as the first and this causes no mistake. This can be found out by a high school student, given there is enough time, appropriate sample problems and careful guidance by the teacher. From this, the algorithm may be formulated quite easily. These observations gave us also a necessary condition for existence

of any solution. Whenever there is no vertice to be chosen (i. e. with no predecessor), there is no task to start with and therefore no solution at all.

This approach may still be refined. Searching for the first vertice (with zero predecessors) is unnecessarily time consuming if don't think it through and do it with no preparation in every step. This is useful to be experienced by students: Having a smart idea leading to a solution undoubtedly, yet not efficiently. At the same time, it is a trap for more advanced students. We want to find the candidates quickly, so one of the ideas could be to sort the vertices according to their incoming degree and keep a sorted list. Actually, a more efficient way can be found, because only the first vertice is needed. Therefore some kind of heap could serve us well. However, heap still means wasting resources, because we know exactly, what of what degree value shall be the first vertice. Finally, it is enough to keep a list of candidates (vertice with no predecessor) and the incoming degree for each vertice. Keeping these two structures up to date can not be faster (we will always need to do at least one operation per vertice and per each of its edges).

The last possibility we describe here is unlikely to be found by students, as it is not very natural. It is based on depth first search (DDS). Therefore it may help to revise DDS. Moreover, this approach may illustrate one of the key concepts in computer science: try to reuse what we already know for solving a new situation. In this case however, it is not straightforward. DDS has to be run on a graph with reversed edges. First it finds vertices with no predecessors (along edges) of this modified graph. Then DFS is run from these vertices one by one. Order of closing vertices is the demanded topological order. If an open vertice is encountered anytime, the original graph contains a cycle and therefore there is no correct topological order. Time complexity of the last two approaches is proportionate to the number of vertices and edges, if appropriate data structures are used.

Further topics

The topic of topological sorting bring an excellent opportunity to seek some further matter - both theory and applications. Orders in general may be introduced, as well as lattices and Hasse diagrams. The strongest topic would probably be deepening planning and scheduling. There is an important counterpart to this out of CS. One of the major areas of project management is basically planning and scheduling without the algorithms. We will mention here one possible and fruitful topic, the critical path method (CPM). For this we need to consider also time consumed by individual tasks, so we add it as weights to vertices.

Finding an order for given set of tasks with their dependencies is certainly useful, yet often insufficient in real life. Tasks can often be accomplished in parallel. This of course does not mean they can be accomplished all at once, there still are their dependencies. A reasonable question in this situation is about the minimum working time to accomplish all tasks. Length of the critical path is the answer. Also, this is a way to define critical path. Looking into it, we would find out it is the longest possible sequence of vertices connected by edges. Critical path is a good example that some useless looking problems, such as searching for the longest path in a graph, may actually be of crucial importance in certain situations.

The total time taken by the project can not be shorter for a simple reason. Each task on the path has to be accomplished, in a time given by its weight. And it has to be accomplished after its direct predecessor and before the direct successor, not along any of them. This holds for all vertices on the path. This shows us the importance of finding all critical paths, not only the length. Should any task on any critical path be delayed, the whole project will inevitably end late. We should remark here that all this is not dependent on the topological order possibly found earlier, CPM only needs the graph itself.

Finding a critical path is surprisingly easy, and therefore suitable for a high school student with a very little computer science experience. We will find the shortest time a vertice may be finished for all the vertices, one by one, using the topological ordering. For a vertice with no

predecessor, the time needed to finish is obviously only the time for the task itself. Whenever we take a vertice in topological order, all its predecessors have been processed already, therefore we know the earliest moment they could be accomplished. The earliest moment to finish the taken vertice one is obviously the time of its last finished predecessor plus the time for the vertice itself. This way, we simply run through the graph and end with an answer at hand. This approach reuses many crucial ideas from topological sorting.

Another way to modify the problem and seek for another solution could be program evaluation and review technique (PERT) or hierarchical tasks using work breakdown structure (WBS). Of course, classically scientific questions may also be solved in the classroom, like what is the total number of all topological orders on given graph.

Summary

In this contribution, we have proposed a new topic for high school education – graph theory. We have examined the current situation, shown why we consider it important and what should be the benefits of including it. Graphs are a very helpful tool for dealing with relations between pairs of objects. This is the way we want high schools students to see graphs. They shall be able to use graphs this way, including some basic traversing and searching. After this we discussed topological sorting in detail. It represents an advanced topic, yet comprehensible for students and moreover, very useful. Many important aspects of graph theory and computer science may be illustrated using this problem. In the last part of this contribution we showed a possible continuation of topological sorting. The critical path method is again a very useful concept, yet comprehensible by students and using some elegant ideas and reasoning.

High school students are already using simple graphs, intuitively. Yet, they are not aware of any special science behind. We believe that showing it to them would improve their lives. They would be able to transfer and use their graph related knowledge and skills in new areas. They would also understand better the limits of drawings and common sense based approach.

References

CERMAT: *Katalog požadavků zkoušek společné části maturitní zkoušky ZKUŠEBNÍ PŘEDMĚT: INFORMATIKA, vyšší úroveň obtížnosti.* Praha: Centrum pro zjišťování výsledků vzdělávání, 2010.

Gal-Ezer, J., Beeri, C., Harel, D., Yehudai, A., A High-School Program, in *Computer Science.* Computer, 1995, 28, 10, pp. 73-80.

Kučera, L.: *Kombinatorické algoritmy*, SNTL, Praha 1983

Lessner, D.: Computer Science Education on High Schools, in *WDS'10 Proceedings of Contributed Papers: Part I - Mathematics and Computer Sciences*, Prague, Matfyzpress. 2010, pp. 110–115.

Töpfer, P.: *Algoritmy a programovací techniky.* Prometheus, Praha 2007.

VÚP: *Rámcový vzdělávací program pro gymnázia.* Praha: Výzkumný ústav pedagogický v Praze, 2007.