

Constructing Knowledge in Graph Theory and Combinatorial Optimization within the multimedia application ‘Graphs’

Eva Milková

eva.milkova@uhk.cz

University of Hradec Králové, Faculty of Informatics and Management
Rokitanského 62, 500 03 Hradec Králové, Czech Republic

Abstract: *Multimedia applications together with individual approaches within the didactic process have substantially influenced education. They give teachers an excellent chance how to support not only demonstrating and visualizing the explained subject matter to be much clearer and comprehensible, but also enable them to prepare such study material for students that optimizes their study habits. The Theory of Graphs together with Combinatorial Optimization is a wonderful, practical discipline. On the one hand, there are many methods which can be used for solving the same problem, while on the other hand, using effective modifications of one algorithm, we can devise methods of solving various other tasks. To educate students in this area it is important to make them familiar with certain algorithms in contexts to be able to get deeper into each problem and entirely understand it. In the paper we present just a few ideas that have proved successful in teaching and learning this part of mathematics with a big support of the program ‘Graphs’, whose main purpose is a visual representation of basic graph-concepts and graph-algorithms using a coloring process on graphs created within the program.*

1 Introduction

Information technology has substantially influenced education. Students find modern technology very handy when looking up things of their own interest. The teachers should take advantage of this fact and should try to prepare for them such multimedia study material, which would optimize their study habits. It means to use applications making students’ study more effective, time-efficient and explained topics more comprehensible.

Along with large software products dealing with a wide spectrum of objects developed by a team of professionals there are also various programs dealing with objects appropriate to course subject matter created on a script given by the teacher with regard to students needs.

This paper illustrates advantages of the program called ‘Graphs’ [1] used as an important support of teaching and learning the subject Graph Theory and Combinatorial Optimization taught at the Faculty of Informatics and Management, Hradec Králové, Czech Republic.

2 Program ‘Graphs’

The program ‘Graphs’ was created in the Delphi 5 environment by our student within his thesis on a script given by the author of this paper. The main purpose of this application is a visual representation of basic graph-concepts and graph-algorithms using a coloring process on graphs created within the program.

The program is divided into three parts: Creation and modification of a graph, Presentation of graph characteristics, Application of several fundamental graph algorithms (see Figure 2.1).

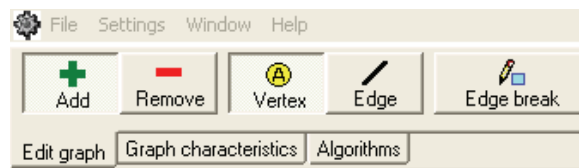


Figure 2.1 Three parts of the program ‘Graphs’ and *Edit graph* menu

The program enables the creation of a new graph, editing it, working on it (moving, coloring vertices, edges, etc.), saving graph in the program, and saving the graph in bmp format. It also makes it possible to display some graph properties of the given graph, to add color to vertices and edges, and to change positions of vertices and edges by “drop and draw a vertex (an edge respectively)”. It allows the user to open more than one window so that two (or more) objects or algorithms can be compared at once.

Main benefit of the program ‘Graphs’ can be characterized as follows.

It enables the teacher to *complete his/her explanation within lectures* in such a way that the topic is more comprehensible:

- The option to open more than one window enables the teacher to **explain the problem from more points of view and show mutual relations among used graph-concepts and algorithms** on graphs.
- The possibility to use colors allows the teacher to **emphasize needed objects and relations**.
- The possibility to save graph in bmt format allows the teacher **easy insertion of needed graphs** (e.g. graphs used during the lecture) **into the study material** (Remark: All figures used in this paper are created within the program ‘Graphs’).

It enables students to *revise subject-matter* and more deeply understand it.

- Students can use not only graphs prepared by the teacher but also graphs created by themselves and **explore the properties of these graphs**.
- The possibility to open more than one window enables students to **follow mutual relations among used concepts and algorithms**.
- The possibility to save graph in bmt format allows students **easy creation needed graphs** for their tasks (texts and/or presentations) where they describe various practical situations with the aid of graphs and solve the given problem.

3 Teaching Principles

The aim of the subject Graph Theory and Combinatorial Optimization is to develop and deepen students’ capacity for logical thinking. Students gain a basic level of competence in graph theory and graph algorithms. Well-prepared students should be able to describe various practical situations with the aid of graphs, solve the given problem expressed by the graph, and translate the gained solution back into the initial situation.

Thus when teaching the subject Graph Theory and Combinatorial Optimization we always lay stress on applying the following principles so that our students, at least most of them, have entirely understand the explained subject matter.

Amusing motivation to the topic

As a motivation to the given topic explained within the subject Graph Theory and Combinatorial Optimization we often use **puzzle**. We make students solve it, discuss their solutions and, after explanation of the appropriate subject matter, we compare students' solutions with **an efficient solution based on the explained matter**.

Teaching in contexts

When we deal with a **particular problem** we try **to examine it from more than one point of view** if possible and **discuss various approaches to its solution**. On the one hand, there are many methods which can be used for solving the same graph-problem, while on the other hand, using effective modifications of one algorithm, we can devise methods of solving various other tasks. In this way students are encouraged to think about each problem more than usual. By comparing the various attitudes students are able to get deeper into the problem and to understand it.

In order to enrich students' view of the discussed subject matter and deepen their awareness of it, we try to **describe a particular problem with real life examples**. We also ask students to give their own examples describing the topic to be sure that they can understand it.

Demonstration and visualization of the particular issue as well as it is possible

“Students need images and visualization in addition to words. Science learning is about creating images in mind, and teaching should support such image formation.”[2]

In the subject Graph Theory and Combinatorial Optimization there is no problem to illustrate the needed concepts on graphs. However, it is very important **to prepare suitable illustrative graphs** and **use colors to emphasize characteristic of the explained concepts**. The colors play also a very important role in explaining graph-algorithms. We **describe them as an edge-coloring or vertex-coloring process**.

Intensification of students' self-preparation

We are very well aware that interesting **study materials prepared for self-study enable students more consistent engagement with the subject**. Students who are familiar with the matter explained during the lecture are good partners and lessons can be run more efficiently, like a discussion or consultation, then.

This was one of the main reasons why the multimedia application ‘Graphs’, visualizing graph concepts and algorithms and supporting preparation of other useful study materials, was created.

The following two case studies demonstrate our teaching methods used in the subject Graph Theory and Combinatorial Optimization keeping the above mentioned principles with the support of the program ‘Graphs’. The first case study illustrates a simple puzzle used as a motivation to the concepts graph and vertex degree and a little complicated puzzle used as a motivation to the concepts subgraph and isomorphism. The second study focuses on the importance of teaching in contexts.

3.1 Case study 1

Competition. This puzzle we use as a motivation to the concepts graph and vertex degree.

An international company opened five positions for translators from the following languages: Russian, German, English, French and Spanish.

Five candidates apply for the job.

Mr. Smith can speak all 5 languages;

Mr. Parker can speak English, French and Russian;

Mr. Thomas can speak German and Russian;

Mr. Brian can speak English and German;

Mr. White can speak Russian and German.

Is the company able to provide all opened positions so that each of candidates would translate just from one language? If it is possible, propose the solution.

Solution. The situation can be easily represented by the following graph.

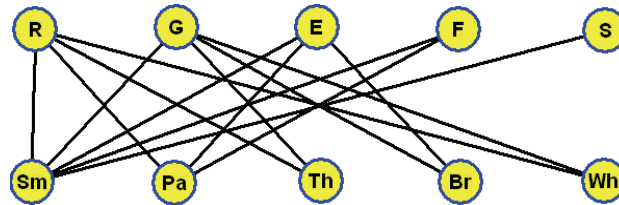


Figure 3.1.1 Graph-representation of the puzzle *Competition*

Looking at the graph on Figure 3.1.1, it is obvious that the vertex S with the degree 1 must be connected with the vertex Sm . Thus, at first, we color the edge $\{S, Sm\}$ blue and the other edges incident with the vertex Sm red to demonstrate that we have rejected them. Then, considering the graph with uncolored edges, it is obvious that the edge $\{F, Pa\}$ must be denoted by the color blue and hence the other edges incident with the vertex Pa by the color red, etc. Finally, the blue edges represent two possible solutions (see the following figures).

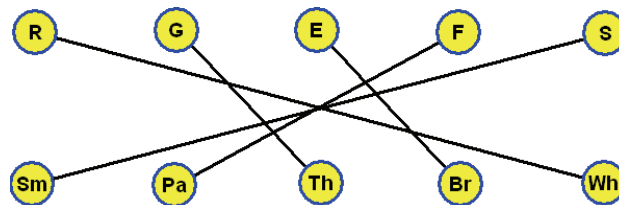


Figure 3.1.2 The first solution of the puzzle *Competition*

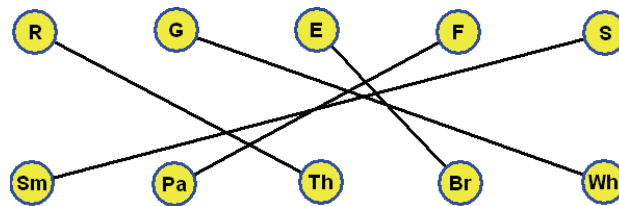


Figure 3.1.3 The second solution of the puzzle *Competition*

We introduce students with the simple puzzle *Competition* during the first lecture when explaining the basic graph concepts. We let students to try to solve the puzzle for 5 minutes. Most of them describe the situation as follows

Smith ... R, G, E, F, S

Parker ... E, F, R

Thomas ... G, R

Brian ... E, G

White ... R, G

and “guess” the puzzle starting with one of the men speaking two languages. In 5 minutes they are usually able to find out only part of the solution and are surprised **how easy it is to solve the given puzzle using graph and colors to visualize needed relations. This process we illustrate using the program ‘Graphs’.**

Much more interesting and also difficult is the following puzzle *Towns* that we discuss with students when explaining the concept isomorphism. Let us introduce it here.

Towns. This puzzle we use as a motivation to the concepts subgraph and isomorphism

Place the names of towns mentioned below into the frames of the given map (Figure 3.1.4) so that there is no same letter in the names of the towns in adjacent frames (neither horizontal nor vertical).

Atlanta

London

Paris

Lima

Berlin

Metz

Quito

Oslo

Caracas

Nairobi

Riga

Tokyo

Dallas

New York

Rome

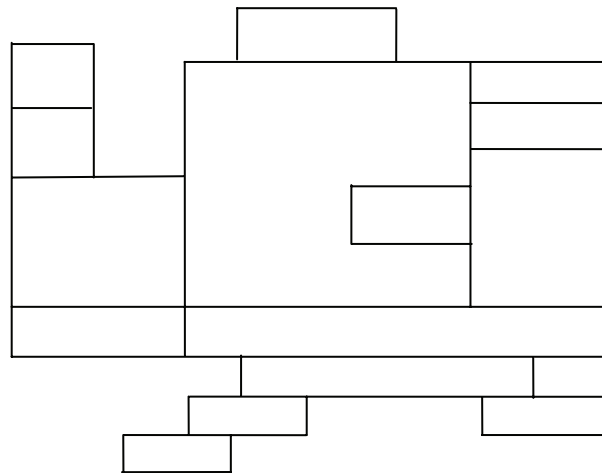


Figure 3.1.4 Map of the puzzle *Towns*

Solution of the puzzle will be presented at the conference.

There is an endless number of enjoyable tasks, puzzles and logic problems in books like “Mathematics is Fun”, in riddles magazines and on the Internet. We have been looking for such that can be efficiently solved with the help of graphs and introduce them in the subject Graph Theory and Combinatorial Optimization on lectures devoted to the appropriate topic.

3.2 Case study 2

From “Jarník to Dijkstra”. Here we introduce a short example of our approach to teaching in contexts. We illustrate a relationship between Jarník’s solution of the Minimum Spanning Tree problem and Dijkstra’s algorithm finding the shortest path between two vertices.

One of the most fundamental well-known problems is the Minimum Spanning Tree (MST in short) problem, which is generally regarded as a cornerstone of Combinatorial Optimization. Moreover, this problem should be interesting for our students also from another reason, namely that two excellent **Czech mathematicians** Otakar Borůvka and Vojtěch Jarník are involved in the genesis of its solutions. First formulation and solution of the problem was given in 1926 by Otakar Borůvka¹ and another solution added Vojtěch Jarník².

(Remark: The survey of the works devoted to the MST problem until 1985 is given in the article [6] and this historical paper is followed up in articles [7] and [8]. Moreover, in the paper [8] it is also shown that out of many available algorithms which solve the MST problem the Borůvka’s algorithm is the basis of the fastest known algorithms)

Let us remind formulation of the problem. In the contemporary terminology MST problem can be formulated as follows (see e.g. [9]):

Given a connected undirected graph $G = (V, E)$ with n vertices, m edges and real weights assigned to its edges (i.e. $w: E \rightarrow \mathbb{R}$). Find among all spanning trees of G a spanning tree $T = (V, E')$ having minimum value $w(T) = \sum(w(e); e \in E')$, a so-called minimum spanning tree.

Here let us describe Jarník’s solution to the MST problem. We describe it as an edge-coloring process (see [9]). The process we demonstrate to students on a connected undirected edge-weighted graph within the program ‘Graphs’.

Jarník’s algorithm

1. Initially all vertices and edges of the graph G are uncolored. Let us choose any single vertex and suppose it to be a trivial blue tree.
2. At each of $(n - 1)$ steps, color the minimum-weight uncolored edge, having one vertex in the blue tree and the other not, blue. (In case, there are more such edges, choose any of them.)
3. The blue colored edges form a minimum spanning tree.

¹ Otakar Borůvka was introduced to the problem by his friend, Jindřich Saxel, an employee of the West_Moravian Powerplants. It was at that time that electrification of the south and west parts of Moravia was beginning, and Borůvka was asked for help in solving the problem. The challenge was how and through which places to design the connection of several tens of municipalities in the Moravia region so that the solution was as short and consequently as low-cost as possible. Borůvka not only correctly stated this problem but also solved it in the papers [3] and [4]).

² Vojtěch Jarník quickly realized the novelty and importance of the problem after reading Boruvka’s paper. However the solution seemed very complicated to him. He started to think about another solution and soon afterwards wrote a letter to Otakar Borůvka in which he suggested a much easier method and consequently he published it in the article [5].

Much later, during the time of newly developing field, computer science, R.C. Prim created the same solution as Jarník. He was unaware of Jarník's solution. He used a more detailed implementation suitable for computer processing in the paper [10].

To use the following description (see [11]) let us consider weights $w(e)$ assigned to edges of a given graph as distances.

Jarník's-Prim's algorithm

1. Let us choose any single vertex a and suppose it to be a blue tree. Put the value $(0, a)$ by the vertex a . By each vertex v which doesn't belong to the blue tree, the actual information $(f(v), u)$ is saved, describing the nearest distance $f(v)$ between the vertex v and the blue tree (from the vertex u). Thus initially by each vertex $v \neq a$ put the value $(w(\{a, v\}), a)$ if v is a neighbor of the vertex a and the value (∞, a) if v is not a neighbor of the vertex a .
2. At each of $(n - 1)$ steps take the following commands:
 - choose a vertex z with the actual information $(f(z), t)$ such that $f(z) = \min\{f(v); v \text{ doesn't belong to the blue tree}\}$,
 - color the corresponding edge $\{z, t\}$ blue,
 - by each neighbor v of the vertex z , change the value $(f(v), u)$ to the value $(w(\{z, v\}), z)$ in the case that $w(\{z, v\}) < f(v)$.
3. The blue colored edges form a minimum spanning tree.

On the lecture we illustrate Jarník's-Prim's solution of MST problem not only **step by step on a graph within the program 'Graphs'** but also in the way described below using adjacency matrix of the given graph.

Let's have the following graph G_1 (Figure 3.2.1) and its adjacency matrix (Figure 3.2.2)

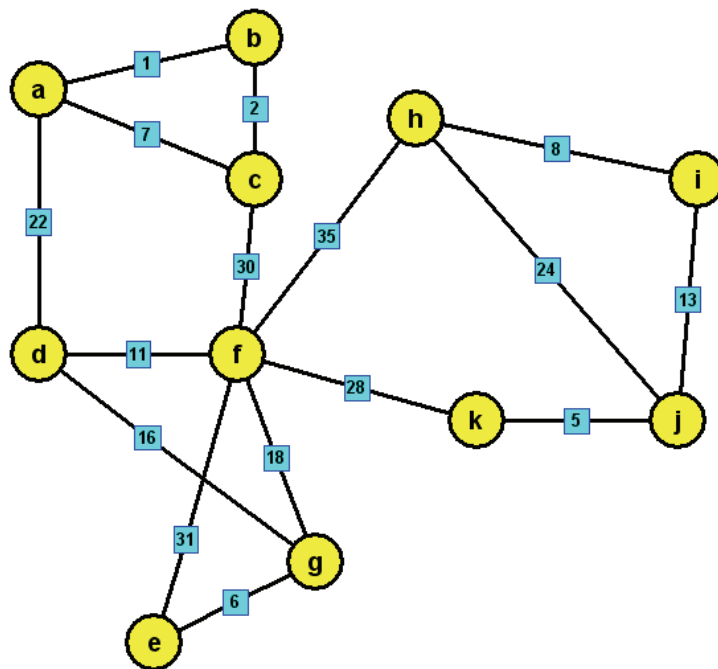


Figure 3.2.1 Graph G_1

	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>G</i>	<i>h</i>	<i>i</i>	<i>j</i>	<i>k</i>
<i>a</i>		1	7	22							
<i>b</i>	1		2								
<i>c</i>	7	2				30					
<i>d</i>	22					11	16				
<i>e</i>						31	6				
<i>f</i>			30	11	31		18	35			28
<i>g</i>				16	6	18					
<i>h</i>						35			8	24	
<i>i</i>								8		13	
<i>j</i>								24	13		5
<i>k</i>						28				5	

Figure 3.2.2 Adjacency matrix of the graph G_1

Jarnik's-Prim's solution of MST problem can be demonstrated as follows:

<i>a</i>	(0, <i>a</i>)										
<i>b</i>	(1, <i>a</i>)										
<i>c</i>	(7, <i>a</i>)	(2, <i>b</i>)									
<i>d</i>	(22, <i>a</i>)	(22, <i>a</i>)	(22, <i>a</i>)								
<i>e</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(31, <i>f</i>)	(6, <i>g</i>)					
<i>f</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(30, <i>c</i>)	(11, <i>d</i>)							
<i>g</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(16, <i>d</i>)	(16, <i>d</i>)						
<i>h</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(35, <i>f</i>)	(35, <i>f</i>)	(35, <i>f</i>)	(35, <i>f</i>)	(24, <i>j</i>)	(8, <i>i</i>)	
<i>i</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(13, <i>j</i>)	
<i>j</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(5, <i>k</i>)	
<i>k</i>	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(∞, <i>a</i>)	(28, <i>f</i>)	(28, <i>f</i>)	(28, <i>f</i>)	(28, <i>f</i>)			

Blue colored edges of the found minimum spanning tree are $\{a, b\}$, $\{b, c\}$, $\{a, d\}$, $\{g, e\}$, $\{d, f\}$, $\{d, g\}$, $\{i, h\}$, $\{j, i\}$, $\{k, j\}$, $\{f, k\}$.

Let's have a question. Could the solution of the MST problem also serve as a solution to the problem of how to find the shortest path from one vertex to another in a connected undirected graph with n vertices and non-negative weights assigned to its edges?

The answer is no. It is easy to see from the figure 3.2.1 that the shortest path from the vertex a to the vertex h in the graph G_1 goes through vertices b, c and f . Its length is 68 while the length of the path (a, d, f, k, j, i, h) between the same vertices in the blue minimum spanning tree found above is 87! However, there is a relationship between the two problems.

Let's consider the following small **modification of the Jarnik's-Prim's algorithm**:

At each step of the Jarnik's-Prim's algorithm, by each vertex v which doesn't belong to the blue tree, save the actual information describing the nearest distance between the vertex v and the initial vertex a (instead the nearest distance between the vertex v and the blue tree).

In this way we get the correct solution, namely the solution found in 1950's by E.W.Dijkstra (see [12]).

At first we illustrate both algorithms on the same graph opened in two windows in the program 'Graphs' and emphasize the difference between them (see Figure 3.2.3).

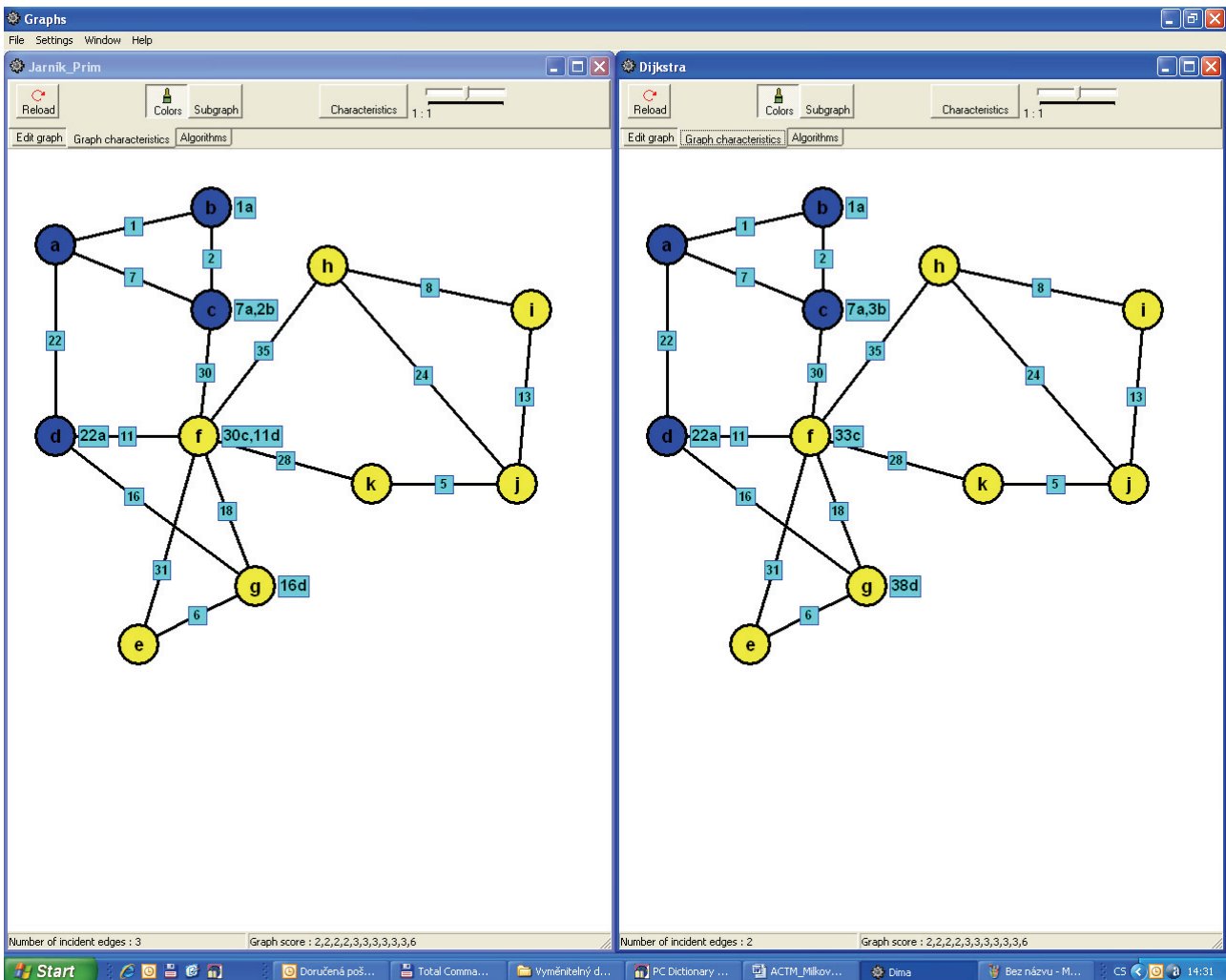


Figure 3.2.3 Third step of Jarnik's-Prim's and Dijkstra's algorithm in two opened window of the program 'Graphs'

Then we let students to solve the Dijkstra's algorithm finding the shortest path between two vertices of a given graph using its adjacency matrix in the way described above, i.e.:

Dijkstra's solution of finding the shortest path from the vertex a to the vertex h in the graph G_1

a	$(0, a)$				
b	$(1, a)$				
c	$(7, a)$	$(3, b)$			
d	$(22, a)$	$(22, a)$	$(22, a)$		
e	(∞, a)	(∞, a)	(∞, a)	(∞, a)	$(64, f)$ $(44, g)$
f	(∞, a)	(∞, a)	$(33, c)$	$(33, c)$	

g	(∞, a)	(∞, a)	(∞, a)	$(38, d)$	$(38, d)$					
h	(∞, a)	(∞, a)	(∞, a)	(∞, a)	$(68, f)$	$(68, f)$	$(68, f)$	$(68, f)$	$(68, f)$	
i	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	$(79, j)$	$(79, j)$
j	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	(∞, a)	$(66, k)$		
k	(∞, a)	(∞, a)	(∞, a)	(∞, a)	$(61, f)$	$(61, f)$	$(61, f)$			

The shortest path from the vertex a to the vertex h in the graph G_1 is (a, b, c, f, h) .

4 Summary and conclusion

Students have all graphs used during the lectures of the subject Graph Theory and Combinatorial Optimization together with the program ‘Graphs’ and other electronic study materials assigned to the subject available in the virtual learning WebCT environment that we use at our university. They can find there also various self-tests with automatic checking.

In the paper we offered a few ideas how to improve teaching and learning the subject Graph Theory and Combinatorial Optimization through technologies while following the main principles mentioned at the beginning. At the conference the program ‘Graphs’ and its use in various examples will be presented.

5 References

- [1] Pozdílek M. (2004). *Grafové algoritmy: vizualizace*. Diploma thesis, Hradec Králové, CZ: University of Hradec Králové.
- [2] Williams, R. (2005). *e-Learning Strategy: What's in the Blend?*. In: Proceedings of the 4th Europ. Conference on e-Learning, Amsterdam, Nederland: ACL, pp. 245-51.
- [3] Borůvka, O. (1926). *O jistém problému minimálním*. Práce Mor. Přírodověd. Spol. v Brně, 3, pp. 37-58.
- [4] Borůvka, O. (1926). *Příspěvek k řešení otázky ekonomické stavby elektrovodních sítí*. Elektrotechnický obzor, 15, pp. 153-154.
- [5] Jarník, V. (1930). *O jistém problému minimálním*. Práce Mor. Přírodověd. Spol. v Brně, 6, pp. 57-63.
- [6] Graham, R. L., Hell, P. (1985). *On the History of the Minimum Spanning Tree Problem*. Annals of the History of Computing 7, 1, pp. 43-57.
- [7] Nešetřil, J. (1997). *A few remarks on the history of MST-Problem*. Archivum Mathematicum Brno, 33, pp. 15-22
- [8] Nešetřil, J., Milková, E., Nešetřilová, H. (2001). *Otakar Borůvka on minimum spanning tree problem: Translation of both the 1926 papers, comments, history*. Discrete Mathematics, 233, pp. 3-36.
- [9] Milková, E. (2007). *THE MINIMUM SPANNING TREE PROBLEM: Jarník's solution in historical and present context*. Electronic Notes in Discrete Mathematics, 28, pp. 309–316
- [10] Prim, R.C. (1957). *The shortest connecting network and some generalization*. Bell. Systems Tech. J. 36, pp. 1389-1401.
- [11] Milková, E.: *Combinatorial Optimization: Mutual Relations among Graph Algorithms*. WSEAS TRANSACTIONS on MATHEMATICS, Issue 5, Volume 7, May 2008, pp.869-879.
- [12] Dijkstra, E. W. (1959), *A note on two problems in connection with graphs*, Numer. Math.,1, pp. 269-271.