NetPro, an Educational Platform for Network Optimization

Dosios K., Paparrizos K., Samaras N. and Sifaleras A.
University of Macedonia, Economic and Social Sciences,
Department of Applied Informatics,
Email: sifalera@uom.gr

Abstract

This paper presents new software for Network Programming, developed for educational purposes. Besides its friendly user interface (GUI), it includes many functions for manipulating Network Programming problems. Our software, NetPro, features implementations of well known algorithms for the solution of shortest paths problems in Acyclic Networks, in nonnegative Cyclic Networks and also in networks with nonnegative arc lengths. Furthermore, there are algorithms for the minimum spanning tree problem, as for example Prim's and Kruskal's. There are also search algorithms which compute in trees or out trees of a directed graph, determine connectivity (strong or not) or produce topological ordering. The user can choose among different data structures. Depending on the function, there is a choice for entering input using Adjacency matrix, or an arc's list or star representations (forward or reverse). Finally, the user has the ability to draw general directed rooted trees, with edges or arcs, in a completely customized way.

Keywords: Network Optimization, Educational Software, Matlab.

1. Introduction

Network Optimization is a core area of Computer Science and Operations Research. It deals with a remarkable range of problems. The reason for that is because many applications from different fields like Engineering, Logistics, Manufacturing, Transportation and Telecommunications can be efficiently modeled using graphs, see [Ahuja et al. (1993)]. During the last years, the most CS curricula contain a course in Graph Theory and Combinatorial Optimization. Not surprisingly, both our undergraduate and postgraduate curriculums feature compulsory courses in Network Programming, too.

Graph algorithms and especially their complexity aspect, is regarded to be a difficult topic in Computer Science. To the best of our knowledge, until now, there was no software capable of implementing graph algorithms. Therefore, the implementation of an educational tool, NetPro, for teaching Graph algorithms was a strong motivation to us. This paper comprises close to the entire population of published Graph Algorithms Software. NetPro is not the only software of this kind. Many other educational or professional programs nowadays exist; each of them, of course, has its own benefits and drawbacks. However, most of the existing software deals with some algorithms and especially with a few network representations.

To begin with the relative work, EVEGA, [Khuri, S. and Holzapfel, K. (2001)] is an educational visualization environment for graph algorithms. EVEGA is an interactive tool, written in the Java language, which implements only maximum flow and preflow – push algorithms. Moreover DIDAGRAPH, [Dagdilelis and Satratzemi (1998)] is software for teaching graph theory algorithms, which allows the user to execute and visualize some

algorithms, step by step. On the other hand, it doesn't provide many varieties in network representations. Furthermore, LEDA (Library of Efficient Data Types and Algorithms), [Mehlhorn, K. and Naher, S. (1995)] is a platform for combinatorial and geometric computing, written in the C++ language. All data types and algorithms in LEDA are precompiled and stored in libraries. Starting version 3.5, the commercial distributor of LEDA is Algorithmic Solutions Software GmbH. CATBox - Combinatorial Algorithm Toolbox, [CATbox], is an interactive course combining a textbook with the visualization software Gato, [Schliep, A. and Hochstattler, W. (1998)], the graph animation tool – box. Gato and CATbox have been developed with Python, [Schliep, A. and Hochstattler, W. (2000)]. Gato is distributed under the GNU Library General Public License (LGPL), while CATbox is to be published by Springer Verlag. Some of the problems covered by CATbox are: minimal spanning tree, shortest-path, maximal-flow and both weighted and non-weighted matching problems.
There are also some books which cover topics about graph algorithms and some of them include also implementations of certain algorithms. Some of them are [R. Sedgewick, (2001A)], [R. Sedgewick, (2001B)] and [R. Sedgewick, (2003)]. Finally Matlog, [NCSU, (2002)], is a Logistics Engineering Matlab Toolbox, which includes functions for graph manipulation and some graph algorithms. Specifically, there are functions for the conversion of one network representation to another as also implementations of Kruskal, Dijkstra algorithms and an algorithm for the minimum cost network flow problem. It doesn't provide any graphical interface to the user, but it features plenty of functions for Optimization problems.

2. The Design of NetPro

NetPro has been implemented using the Matlab language. The current version of Matlab is quite sophisticated and provides the user with many useful tools and functions, see for example [Paparrizos, K. (2001)]. Furthermore, Matlab combines also graphics and animation, see for example [Marchand, P. and Holland, O., T. (2002)] and give the user, the ability to deploy graphical user interfaces (G.U.I.) easily.
All the functions in *NetPro* are separate m-files which have been named characteristically after the respective algorithms. Their names are often constructed as two parts. The first part denotes the respective algorithm, while the second part denotes the network representation used. In order to clarify the naming convention used, it could be mentioned for example the function *kruskal_Adj.m*. This specific m-file obviously implements the Kruskal algorithm using the Adjacency matrix for input. Another function is the *Kruskal_list.m*, which implements the Kruskal algorithm, but the difference in this case, is that a list of arcs is being used.
A large set of Matlab m-files is needed to implement the chain of functions calls for all algorithms and problem sets. *NetPro* features about twenty algorithms, while all of them have open source code. This fact constitutes a benefit for the user, because it gives the potential to the user – student to verify the correctness. Moreover the user can change the code. The installation in Matlab is an easy process and can be done as with any other function or toolbox. All the necessary files must be copied to a folder like Matlab / Work and consecutively the user must select *File* ⇨ *Set Path* ⇨ *Add* with subfolders, to add *NetPro* into the current path of Matlab. Besides the figures and the menu bar the user may use the functions like any other ordinary Matlab function through the Matlab command

line. In every m-file there are specific details about the current input or for any other parameter.

The process of optimization in *NetPro* is described in figure 1.

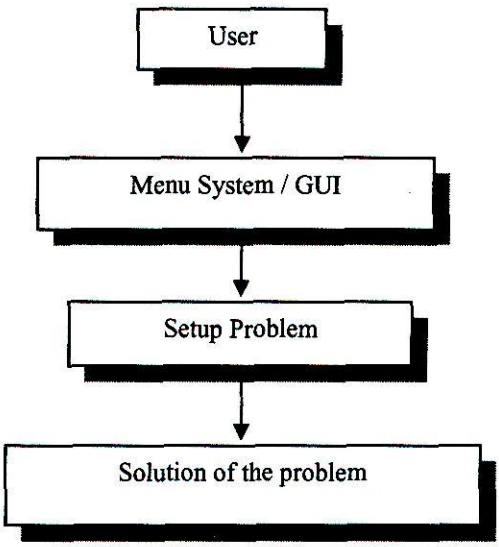

Figure 1. NetPro Optimization Process

During the 1st phase, the user may enter the necessary data input of the problem. Among the problems which can be solved the user can find Shortest Path problems, Minimum Spanning Tree problems, or he may even search a network for certain attributes like Connectivity (strong or not) or Directed Cycles. The user can also transform one network representation to another and choose between Adjacency matrix, a list of arcs or star representation. In the 2nd phase, *NetPro* solves the problem and presents the output to the user in the same figure. This way, the student can change the input e.g. arc's weight and interactively change the output.

At this point, a full example of the solution of a shortest path problem in a network with nonnegative arc lengths is going to be given. As it is known from the theory, Dijkstra's algorithm can solve efficiently the above problem. We assume that the network of figure 2 is given.

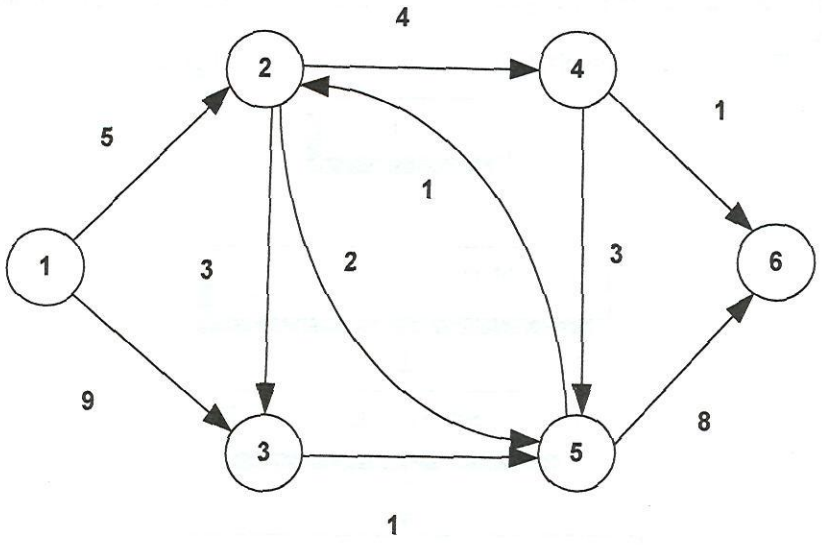

Figure 2. *Dijkstra's Algorithm's example.*

The nodes can be regarded as cities, while the arcs can be regarded as one direction roads between them. The numbers next to the arcs denote the length of the roads. Suppose, for example, that a company wants to transfer certain amounts of its products between different cities. The products are in the 1st city (supply node) and must be transferred the sooner to all the rest cities.

In order to solve the previous problem with *NetPro*, the user has two choices. The first is to solve the problem using the command line. To start with, he may use the function *Dijkstra_Adj.m*, the name of which reveals that implements the Dijkstra algorithm using the Adjacency matrix. The inputs for this function are the Adjacency matrix, a second matrix in which the arc weights are stored and the starting node. The outputs of this function are two vectors of the same size as the number of cities. The first vector keeps the minimum distances of the cities from the supply - city. All the places in the vector denote a city having the same number. The second output – vector keeps the "parent" city for every city visited. The user must give the input through the command line:

A = [0 1 1 0 0 0; 0 0 1 1 1 0; 0 0 0 0 1 0; 0 0 0 0 1 1; 0 1 0 0 0 1; 0 0 0 0 0 0];

W= [0 5 9 0 0 0; 0 0 3 4 2 0; 0 0 0 0 1 0; 0 0 0 0 3 1; 0 1 0 0 0 8; 0 0 0 0 0 0];

The Adjacency matrix representation of a graph is a V – by – V array of Boolean values, with the entry in row *u* and column *w* defined to be 1 if there is an edge connecting vertex *u* and vertex *w* in the graph, and 0 otherwise. In this example, A is the following matrix:

$$A = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

The second matrix W, is the same as the Adjacency matrix, but it keeps the arc's weights in the same places where the Adjacency matrix had values equal to 1. In this example, W is the following matrix:

$$W = \begin{bmatrix} 0 & 5 & 9 & 0 & 0 & 0 \\ 0 & 0 & 3 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 3 & 1 \\ 0 & 1 & 0 & 0 & 0 & 8 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Now the user may type:

[d, p] = Dijkstra_Adj(A,W,1)

The third input 1, denotes the city (starting node) from which all the distances from the remaining cities are going to be calculated. The outputs are the following:

d = [0 5 8 9 7 10]
p = [1 1 2 2 2 4]

For example, the 2nd city is at a distance of 5 (d(2) = 5) from the 1st city and the products arrive there from the first city (p(2) = 1). Similarly, the 4th city is at a distance of 9 (d(4) = 9) from the 1st city while the products arrive there from the second city (p(4) = 2).
In the next section, the second choice for the user will be described; to solve the above problem using the graphical menu.
Finally it must be mentioned that there is also a help file in HTML format, which describes scholastically and through detailed examples all the different menu choices.

3. The Graphical User Interface (G.U.I.) of NetPro

NetPro makes use of all the capabilities of modern technology, which are offered from the Windows operational system and Matlab version R13. The user can start *NetPro* easily, by calling the function *netpro.m* through the Matlab environment. The Graphical User Interface consists of a main menu bar, which itself consists of several other pop up menus, as it is illustrated in figure 3.

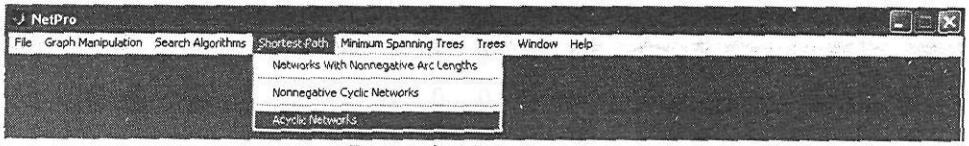

Figure 3. *Main menu bar.*

NetPro is a friendly application, for the Operations Research Scientist, as also for the novice user – student. It performs quickly and efficiently the task which is scheduled to carry out, while on the other hand its graphical environment can easily be learned by the user.

Now, the solution of the previous problem, via the graphical menu, will be given. The user must first start *NetPro,* and then from the menu bar he must select *Shortest Path* → *Networks With Nonnegative Arc Lengths*. At this point the following window, in figure 4, will demand from the user to enter his, or her, preference:

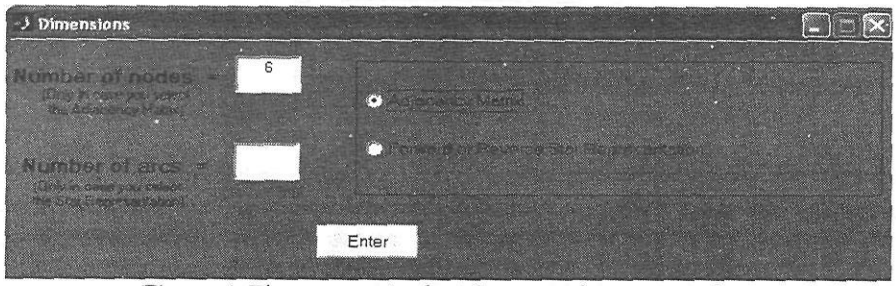

Figure 4. The user must select the network representation.

In this example, the user must select the first choice, as indicated above. After this step, a new window, in figure 5, will demand from the user to fill the two matrixes mentioned previously and to set the starting node.

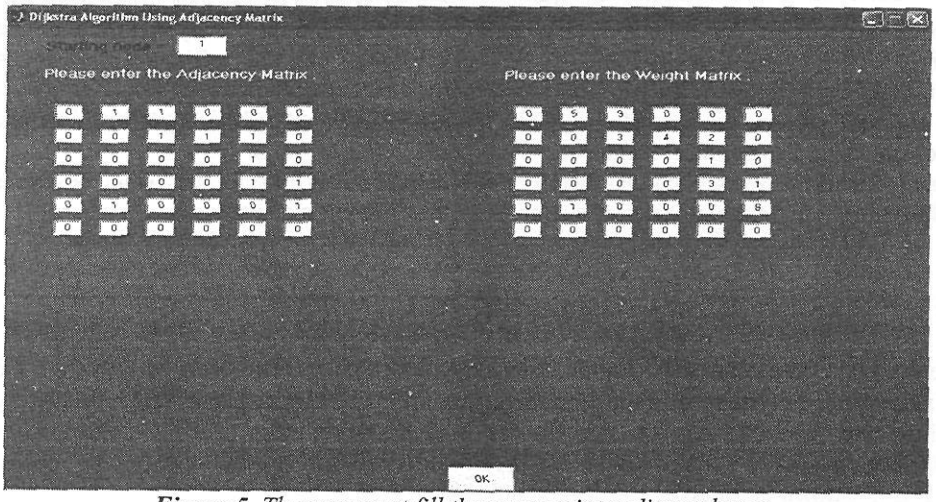

Figure 5. *The user must fill the appropriate edit text boxes.*

The outputs are going to be presented in the same figure as the input, but are only visible to the user, after the user enters the necessary data. In figure 6, anybody can see both the input and the output – solution.

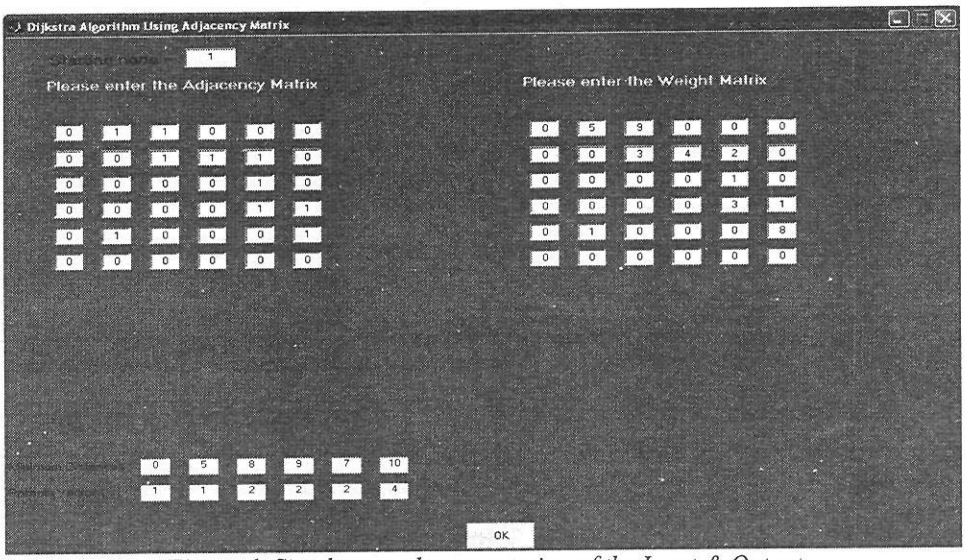

Figure 6. *Simultaneously presentation of the Input & Output*

The two vectors, mentioned in the previous section, are presented at the bottom of the figure in order for the user to enter matrices of larger dimensions. The user may change all the filled values in the matrixes or the starting node, and the outputs are going to change also instantly. This way the user – student can experiment himself via the trial and error method. Even if the user can't enter large dimension problems, for example 20x20, he can still solve the problem through the command line.

Finally the user can draw precisely any kind of rooted tree; directed or undirected. There is also other graphical software which can plot trees, as for example [Herman, I. et al. (1999)].The user can fully customize the tree drawing and may select among different node shapes or colours. It is obvious that instructors can draw any kind of tree easier than they do at the traditional classroom teaching of trees. In figure 7, anybody can see an example of drawing a tree with 19 nodes, using yellow circles as preferred colour and shape for the nodes.

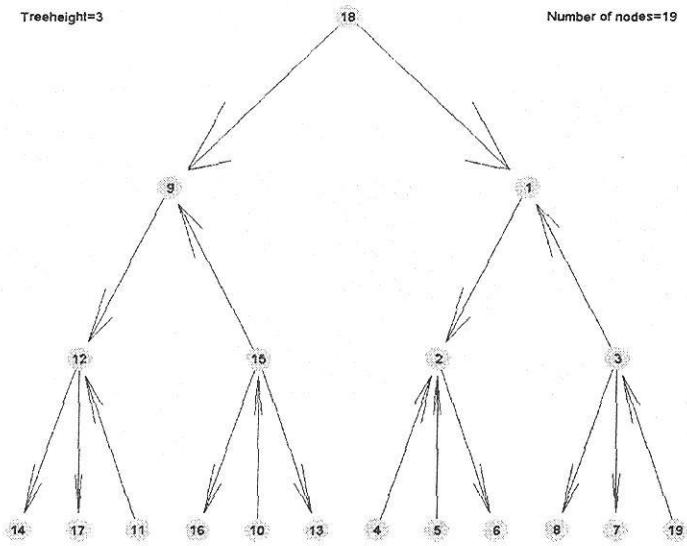

Figure 7. *Tree drawing*

4. Conclusions and Future works

NetPro constitutes a powerful tool; it is user - friendly and has a flexible Graphical User Interface. Its capabilities are making it attractive to many fields of Operations Research like the Decisions Support Systems, but mainly in the teaching of Network Programming. The design of *NetPro* is completely customized and can be implemented in various fields.

NetPro, is going to be tested thoroughly inside the classrooms during the lectures, for the needs of our Department's courses. It is more significant to find how students use this specific software, rather than just explaining what NetPro shows them, [Hundhausen, D. C. et al. (2002)]. Other studies show how to use algorithm animations in learning situations, as for example [C. Kehoe et al. (2001)]. This is the only way for measuring the effectiveness of any software dealing with Algorithm Visualization, described by [B. A. Price et al. (1993)] as a subclass of software visualization, concerned with illustrating computer algorithms in terms of their high-level operations, usually for the purpose of enhancing computer science students' understanding of the algorithms' procedural behavior. Besides, there have been conducted researches which support that pedagogical advantages can be only partially attributed to Algorithm Visualization Technology, as for example [M. D. Byrne et al., (1999)].

A future goal is to make NetPro independent from the Matlab language and the Microsoft Operational System. Finally, another possible direction for future development is to integrate *NetPro* and *LinPro* [Dosios, K. et al. (2002)]. *LinPro,* is implemented in the Matlab language and it is an educational platform for Linear Optimization. Both the packages can constitute a complete Optimization suite, useful in Research or Education in many fields.

References

Ahuja, K. R., Magnanti, L., T. and Orlin, B., J. (1993). Network Flows: Theory, Algorithms and Applications. Published by Prentice Hall, Englewood Cliffs, NJ.

M. D. Byrne, R. Catrambone and J. T. Stasko, (1999). Evaluating animations as students aids in learning computer algorithms. Computers and Education, vol. 33, 253 – 278.

CATBox - Combinatorial Algorithm Toolbox. URL: http://www.zpr.uni-koeln.de/~catbox , to be published by Springer Verlag.

Dagdilelis, V. and Satratzemi, M. (1998). DIDAGRAPH: Software for Teaching Graph Theory Algorithms. Proceedings of ITiCSE 98, Dublin, Ireland, 64-58.

Dosios, K., Paparrizos, K., Papatzikos, N. and Sifaleras, A. (2002). LinPro, an Educational Informational System for Linear Programming. To be published in Proceedings of 15th Pan-Hellenic Conference of Greek Operations Research Society, Tripoli.

Hundhausen, D. C., Douglas, A., S. and Stasko, T., J. (2002). A Meta-Study of Algorithm Visualization Effectiveness. Journal of Visual Languages and Computing, vol. 13(3), 259-290.

Herman, I., Melancon, G., De Ruiter, M., M. and Delest, M. (1999). Latour - a Tree Visualization System. In proceedings of International Symposium on Graph Drawing GD'99, published by Springer – Verlag.

C. Kehoe, J. T. Stasko and A. Taylor, (2001). Rethinking the evaluation of algorithm animations as learning aids: an observational study. International Journal of Human-Computer Studies, vol. 54, 265-284.

Khuri, S. and Holzapfel, K. (2001). EVEGA: An educational visualization environment for graph algorithms. ACM SIGCSE Bulletin, vol. 33(3), 101-104.

Marchand, P. and Holland, O., T. (2002). Graphics and GUIs with MATLAB, 3rd. Published by CHAPMAN AND HALL/ CRC.

Mehlhorn, K. and Naher, S. (1995). LEDA: A platform for combinatorial and geometric computing. Communications of the ACM, vol. 38(1), 96-02.

NCSU. (2002). Matlog Version 6. URL: http://www.ie.ncsu.edu/kay/matlog.

Paparrizos, K. (2001). MATLAB 6.0. Published by ZYGOS Publications, Thessaloniki.

B. A. Price, R. M. Baecker and I. S. Small (1993). A principled taxonomy of software visualization. Journal of Visual Languages and Computing, vol. (4), 211 – 266.

R. Sedgewick, (2001A). Algorithms in C, Part 5: Graph Algorithms, 3rd Edition, Published by Addison-Wesley, Boston.

R. Sedgewick, (2001B). Algorithms in C++, Part 5: Graph Algorithms, 3rd Edition, Published by Addison-Wesley, Boston.

R. Sedgewick, (2003). Algorithms in Java, Part 5: Graph Algorithms, 3rd Edition. Published by Addison-Wesley, Boston.

Schliep, A. and Hochstattler, W. (1998). Gato - Graph Animation Toolbox. URL: http://www.zpr.uni-koeln.de/~gato.

Schliep, A. and Hochstattler, W. (2000). Developing Gato and CATBox with Python: Teaching graph algorithms through visualization and experimentation. In proceedings of Multimedia Tools for Communicating Mathematics (MTCM'00).