# A web-based solver for the single source shortest path problem

Baloukas Thanasis , Badr El-Said[1] and Paparrizos Konstantinos
Dept. of Applied Informatics, University of Macedonia
156 Egnatia Street, 54006 Thessaloniki
{thanasis, it02185, paparriz} @ uom.gr

**Abstract**. In this work we present a new web-based visualization software which has two uses: it can be used either as solver for the single source shortest path problem or as teaching and self-studying material for common shortest path algorithms. The software features running animations for many graph and network algorithms. Among others it encompasses animations for three well-known graph algorithms: the DAG shortest paths algorithm which applies in weighted directed acyclic graphs, the Dijkstra's algorithm for graphs with nonnegative weights and finally the Bellman-Ford algorithm that handles negative weights and can also detect negative cycles. The proposed tool can be used by teachers trying to explain the abovementioned algorithms to students. In addition it has been programmed as Java applet and is therefore freely accessible from the web by any student who wants to deepen their understanding of shortest paths algorithms. The use of visualization in teaching algorithms courses is being used extensively the last decades. As a consequence, many similar tools have been developed. We argue how the proposed software overcomes limitations exhibited from similar algorithm visualization tools. Finally we document our experiences using it in the Network Optimization course at our Department and we conclude with future research this work may lead.

**Keywords.** Operational Research education, Network Optimization, Graph Algorithms, Algorithm Visualization, Distance Learning.

## 1. INTRODUCTION

Algorithm visualization (AV) or animation appeals to the strengths of human perception by providing a visual representation of the data structures and by allowing for smooth image transitions between time-related visual representations that correspond to different states of the execution of the algorithm. In order to improve the teaching quality, many tutors are using AV software in their courses. The execution of an algorithm is inherently dynamic, so it is often hard to be explained using the blackboard or the whiteboard. For network optimization courses, the use of visualization is natural because of the convenient graphical representation of nodes as points and edges as lines between pairs of nodes.

The AV tools that have been developed hitherto can be categorized into two groups. Many have students watch already prepared animations. An alternative strategy is to have the students themselves construct animations of algorithms The AV tool we propose falls into the latter group.

At this point it should be noted that the software being discussed apart from algorithm animations for the single-source shortest paths problem, encompasses also animations for Graph Algorithms such as Search, Minimum Spanning Tree and Network Flow algorithms.

[1] Research supported by the Greek Scholarship Foundation (I.K.Y)

More over our didactic tool has the following characteristics:

- It can be run either remotely as Java [4, 6, 8, 12 and 16] applet or locally as standalone Java application.
- It is highly interactive, including a graph editor for creating new graphs and modifying existing ones.
- It allows students to provide their own input to the algorithm being animated.
- It allows students to interactively run visualizations either at once or step by step, including rewind facilities.
- It presents to the students textual explanation at the same time with the visual display.
- It demonstrates algorithm visualizations for directed and undirected graphs.

For a comprehensive description of the tool the reader is referred to [3]. The proposed software is freely accessible and can be run as Java applet at http://eos.uom.gr/~thanasis/18eeee.htm. Moreover it can be executed locally as standalone Java application, after downloading a single executable file from the link: http://eos.uom.gr/~thanasis/18eeee.jar .

## 2. RELATED RESEARCH

Increasingly many AV tools have been developed in the past two decades.

The Network-enabled solver [2], is an interesting Java application that features only a visualization for the "Achatz, Kleinschmidt and Paparrizos" algorithm for the Assignment Problem.

PILOT [5], is a web-based interactive system for testing computer science concepts. In addition, it demonstrates visualizations for several graph algorithms. Besides, it can test algorithmic concepts, it supports graph generation and layout and finally it incorporates an automated grading mechanism. Nonetheless it creates only random instances of a problem and it doesn't include any graph editor.

JIVE [7], is a system for the visualization of Java coded algorithms and data structures. It treats the algorithms and their visualizations as two distinct entities that interact by transparently using a generic communication channel. It can be used for algorithm debugging as well as in Virtual Classroom environments. In addition, it provides for the visualization of both small and large data sets using semantic zooming. On the other hand, it doesn't have rewind facilities and it only encompasses visualizations for the dfs algorithm, the minimum vertex cover problem and the graph coloring problem.

EVEGA [10], is an educational visualization environment for graph algorithms. It incorporates a convenient graph editor where one can easily create and edit graphs. The user can start/stop the visualization and he/she can change its speed. Nevertheless it doesn't offer stepping back conveniences, it doesn't support visualizations for undirected graphs and in its current version it only features visualizations for the bfs, the dfs and a push/relabel algorithm for the maximum flow problem.

The educational tool described in [11] is a Java application that only features a visualization for the revised simplex method.

JHAVE [13] is a client-server architecture, implemented in Java, into which more specific AV engines may be plugged. JHAVE presents students with two types of visualizations: smoothly running animations and sequences of discrete snapshots. Moreover, it supplements the visualization with context-sensitive textual material and it allows students to provide input to the algorithm being animated. On the other hand, as it is currently configured, its documentation

it doesn't produced dynamically with algorithm's execution. What is more, it doesn't include any graph editor.

ANIMAL [15] is a new tool for developing animations of algorithms, data structures and many other topics. Animations are generated using a visual editor, by scripting or via API calls. It supports source and pseudo code inclusion and highlighting as well as precise user-defined delays between actions. Nonetheless it doesn't provide any graph editor in order for the user to create/modify a graph and it doesn't allow the user to try out different inputs for the algorithm being animated.

DisViz [17] is a distributed visualization tool designed to assist students in learning graph algorithms. It is intended for collaborative use by a group of students over a classroom network. It supports a variety of classroom uses: visualization of graph algorithms, case study in an operating systems class and distributed application development. However the user cannot see a running animation step by step and he/she cannot provide various input data sets for the algorithm being animated.

Taking into consideration the tools mentioned above we argue that the combination of characteristics the proposed software exhibits, **cannot be found** in other similar purpose visualization software.

## 3. THE SINGLE SOURCE SHORTEST PATH PROBLEM

The single source shortest path problem, concerns finding the shortest paths from a specific source node (s) to every other node in a weighted directed graph [1, 9, 14]. The DAG shortest paths algorithm solves the problem stated above in weighted, directed acyclic graphs. Dijkstra's algorithm solves this if all weights are nonnegative. The Bellman-Ford algorithm handles any weights. Moreover it can detect negative weight cycles. We consider as source node the node labeled with 1. For a detailed description of the problem as well as of the algorithms that solve it the reader is referred to [1, 9 , 14].

### 3.1. An animation example. The Bellman-Ford Algorithm

At this time we mention the Bellman-Ford Algorithm in pseudo code format.

**Input**: the graph **G**, the source node **s**, the arcs weight vector **w**
**Output**: **d,** vector of shortest paths for all nodes u reachable from s
**p,** vector of parent pointers (fathers) in shortest path tree

```
d(s) ← 0
d(u) ← ∞ , for each node u ≠ s
p(u) ← nil, for each node u ϵ N
for i← 1 to n − 1 do // the graph has |N|=n nodes and |A|= m arcs
   for each arc (u, v) do
      if  d(v) > d(u) +w(u, v)
          d(v) ← d(u) + w(u, v)
          p(v) ← u
      end_if
   end_for
end_for
```

**for** each edge (u, v) **do**
  **if** d(v) > d(u) + w(u, v)
    Return FALSE //negative weight cycle
  **end_if**
Return TRUE

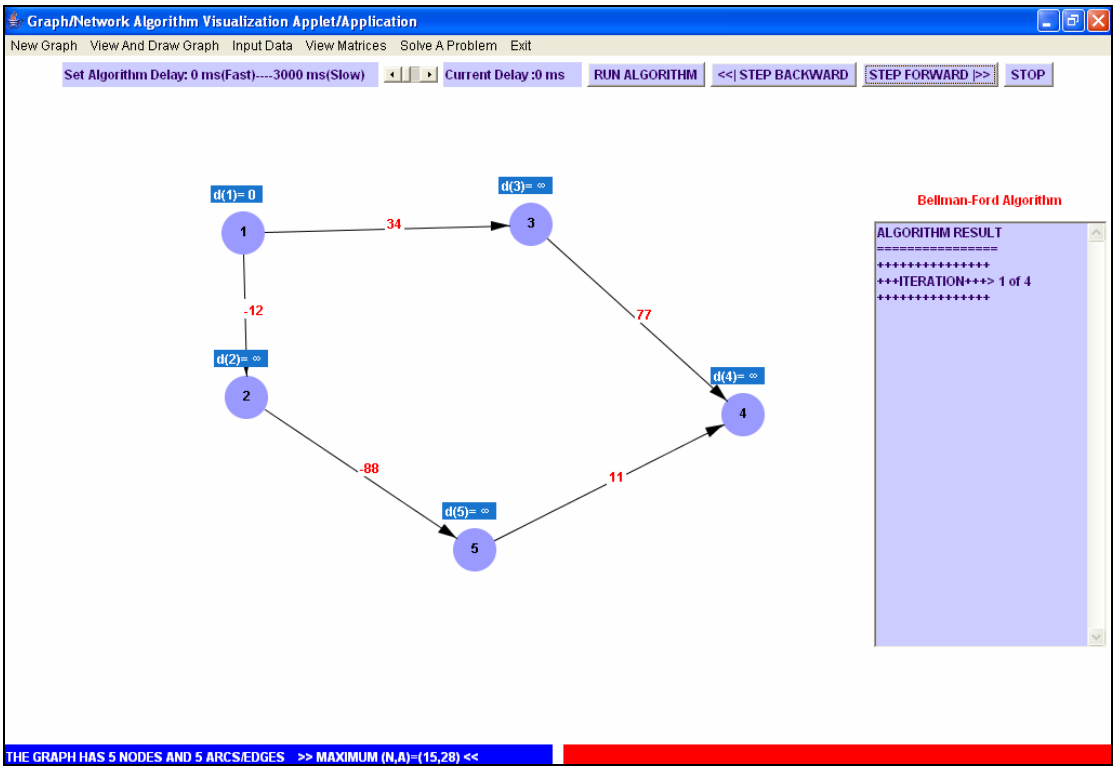


**Figure 1.** Initialization of algorithm.

In figure 1 a directed graph having negative weights in arcs is illustrated. In addition the algorithm is initialized; d(1) is set to 0 and d(v) is set to ∞ for all other nodes.

In figures 2 and 3 the algorithm execution is depicted just after the user having pressed the step button 5 and 6 times respectively. We notice how node 4 changes its father in the shortest path tree; from node 3 in figure 2 to node 5 in figure 3.

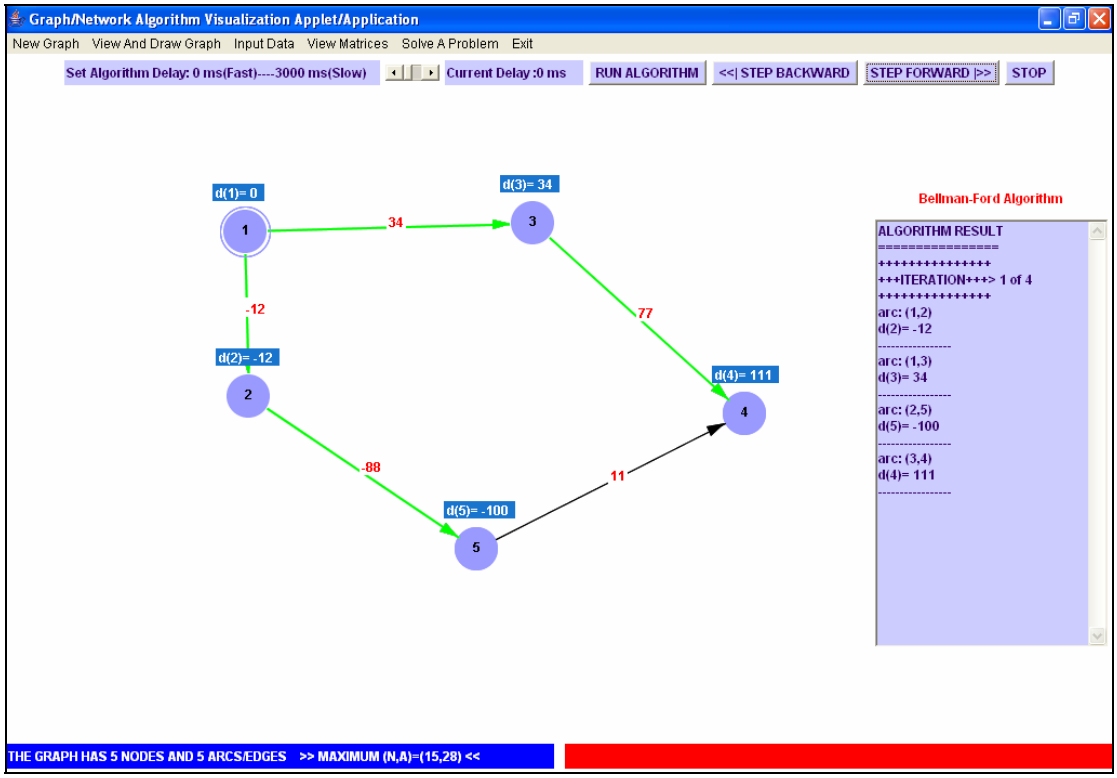


**Figure 2.** Algorithm execution at step 5.

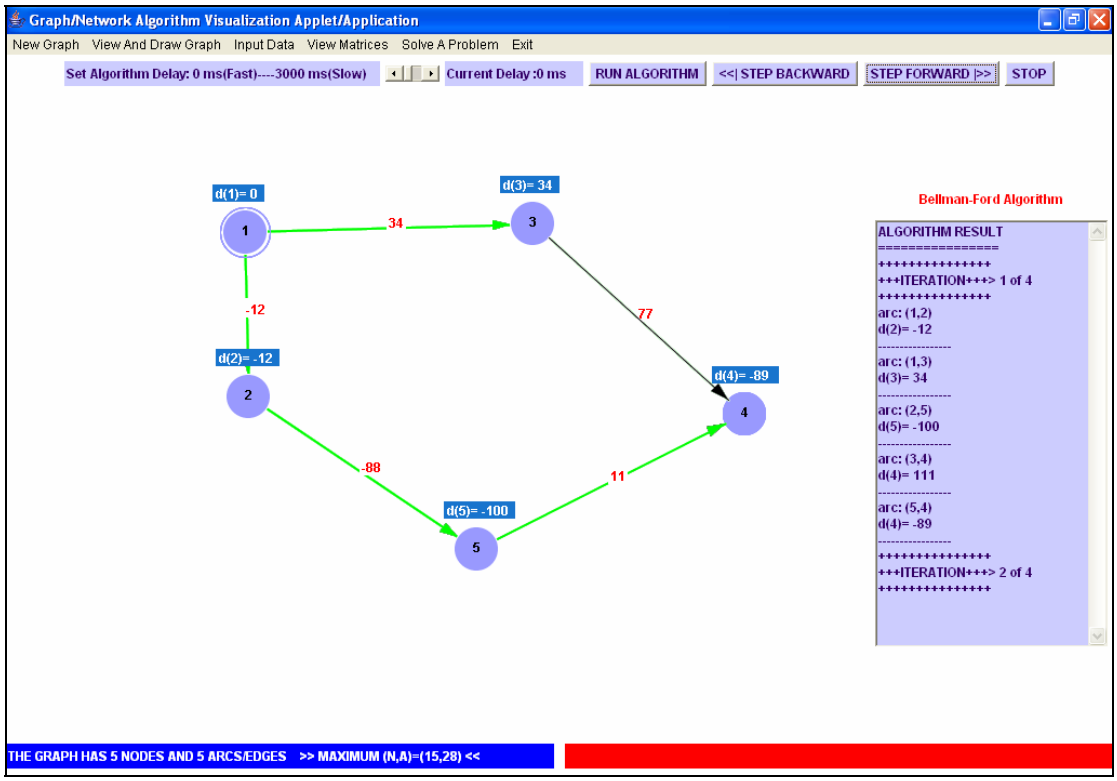


**Figure 3.** Algorithm execution at step 6.

## 4. FUTURE RESEARCH

We have already demonstrated the proposed tool to students, in the course network optimization. There were positive comments and most students liked the GUI of the tool. In addition they liked our choice of colors we employed for the visualization of algorithms.

Recent research has been conducted to empirically evaluate the pedagogical value of students viewing and interacting with prepared algorithm animations. Results have provided mixed results regarding the educational usefulness of similar visualization tools. Many studies argue that students only slightly benefit from animations. Other studies [19] claim that similar tools don't assist in learning as much as previously thought. Motivated by such studies, we are going to assess the learning value of the proposed tool during the spring semester at the Department of Applied Informatics of University of Macedonia.

## References

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

[2] Andreou D. et al. (2005), "Application of a New Network-enabled Solver for the Assignment Problem in Computer-aided Education", *Journal of Computer Science*, 1(4), pp. 19-23.

[3] Baloukas Th., Paparrizos K. and Rossiou E. (2005), "Open and distance learning endorsement through a web-based visualization software", in Proceedings of the Third International Conference on Open and Distance Learning 'Applications of Pedagogy and Technology' (ICODL '05), Patra, Greece, pp. 341-349.

[4] Bates B. and Sierra K. (2003), "Head First Java", O'Reilly.

[5] Bridgeman S., Goodrich M., Kobourov S. and Tamassia R. (2000), "PILOT: an interactive tool for learning and grading", in Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, Austin, Texas, United States, pp. 139-143.

[6] Campione M. et al. (1988), "The Java Tutorial Continued: The Rest of the JDK", Addison-Wesley Professional; Bk & CD-Rom edition.

[7] Cattaneo G., Faruolo P. and Petrillo U. (2004), "JIVE: Java Interactive software Visualization Environment", in Proceedings of the 2004 IEEE Symposium on Visual Languages and Human Centric Computing (VLHCC'04), Rome, Italy, pp. 41-43.

[8] Cohen S. et al. (1996), "Professional Java Fundamentals", Wrox Press.

[9] Goodrich M. and Tamassia R. (2001), "Algorithm Design: Foundations, Analysis, and Internet Examples", John Wiley & Sons, Inc.

[10] Khuri S. and Holzapfel K. (2001), "EVEGA: an educational visualization environment for graph algorithms", in Proceedings of the 6th annual conference on Innovation and technology in computer science education, Canterbury, UK, pp. 101-104.

[11] Lazaridis V., Samaras N. and Zissopoulos D. (2003), "Visualization and Teaching Simplex Algorithm", in Proceedings of the 3rd IEEE International Conference on Advanced Learning Technologies (ICALT'03), Athens, Greece, pp. 270-271.

[12] McBride P. (2002), "Java Made Simple", Made Simple Series.

[13] Naps T., Eagan J. and Norton L. (2000), "JHAVE- An environment to actively engage students in Web-based algorithm visualizations", in Proceedings of the 31st SIGCSE Technical Symposium on Computer Science Education, Austin, Texas, United States, pp. 109-113.

[14] Paparrizos K., Samaras N. and Sifaleras A. (2005), "Network Optimization", University of Macedonia Press.

[15] Robling G., Schuler M. and Freisleben B. (2000), "The ANIMAL Algorithm Animation Tool", in Proceedings of the 5th annual SIGCSE/SIGCUE ITiCSE conference on Innovation and Technology in Computer Science Education, Helsinki, Finland, pp. 37-40.

[16] Sikora Z. (2003), "Java Practical Guide for Programmers", Morgan Kaufmann.

[17] Sherstov A. (2003), "Distributed Visualization of Graph Algorithms", in Proceedings of the 34th SIGCSE technical symposium on Computer science education SIGCSE '03, Reno, Nevada, US, pp. 376-380.

[18] Stasko J., Badre A. and Lewis C. (1993), "Do algorithm animations assist learning? An empirical study and analysis", in Proceedings of the SIGCHI conference on Human factors in computing systems, Amsterdam, The Netherlands, pp. 61 – 66.