Teaching Graph Algorithms Using Online Java Package IAPPGA

Dr. Mingshen Wu
Department of Mathematics, Statistics, & Computer Science
University of Wisconsin-Stout, Menomonie, WI 54751
E-mail: wuming@uwstout.edu

Abstract
Teaching and learning graph algorithms is a great challenge to both instructors and students. Instructors are seeking software that is specifically designed to demonstrate the algorithms and for students to learn these algorithms efficiently. The software program should be readily available and provide an environment so that students are able to review the algorithm, solve a practical problem, and intuitively study the working process via a graphical display all together. This paper presents an Internet Accessible Program Package for Graph Algorithms (IAPPGA) developed by the author. This package can be accessed via an Internet browser at any time, anywhere without downloading or installing any software.

Keywords
Graph, Algorithm, Online, Drawing

1. Introduction

Almost all universities/colleges teach graph algorithms in a math course and/or a computer science course. A fundamental activity of applying graph theory is the ability to learn and implement scientific algorithms on a computer. This is a challenge for most students enrolled in a course that studies graph algorithms because traditional instruction (simply introducing the algorithms in the classroom) does not help students to apply algorithms that they have learned. Drawing graphs certainly is an important way of learning graph theory. That is why many people are interested in developing the skill of drawing graphs. One can learn how to draw graphs via a workshop such as the one at the MAA-AMS joint meeting (January 5-8, 2005, Atlanta) that provided a chance to learn graph drawing using Flash. It would be nice if students could have a software package that provides the option of drawing graphs online and visualizing the algorithms while studying graph algorithms. The graphical feature is very helpful to enhance students' understanding of algorithms and their implementations on a computer. A typical feature of graph algorithms is optimization such as finding the shortest distance of a transportation network, the best job schedule, a minimum spanning tree, or the maximum flow of networks. These algorithms perform certain steps to achieve the final answer. Thus, visualizing the working process is the most efficient way to help students.

Motivated via teaching experience, the author started to develop an Internet Accessible Program Package for Graph Algorithms (IAPPGA) several years ago. The algorithms were selected from the textbooks [1], [2], and [7]. The purpose of developing IAPPGA is to provide students a convenient environment to study graph algorithms and to provide the instructor a tool for teaching and demonstrating graph algorithms. Several years ago, I asked my former colleague, Prof. Stuart Hansen to check my work and comments. I learned that Dr. Hansen was developing software that visualizes graph algorithms for teaching Java language during the same period of time [3]. Based on different purposes, we independently worked on the visualization of graph theory algorithms. I also visited quite a number of web sites to find out other valuable work in this area. Comparing IAPPGA with some existing work such as Graph Magic [3], JGraphED [4], and EVEGA [5], IAPPGA has three significant features: 1. This package can be accessed via a web browser without downloading or installing any software; 2. The graphical display of the graphs a user inputs (either by online drawing or by an adjacency matrix) is completely adjustable, that is, the user can freely change the shape of the graphs to study the solution via the graphical display during executing an algorithm; and 3. Besides the text display and graphical display of the solution process, this package also provides students an online review of the key idea for each algorithm.

Below, section two introduces the IAPPGA, section three discusses the impact to teaching and learning, and section four is the conclusion.

2. The IAPPGA package

This package provides for individual learning at a pace appropriate for the student. It allows time for learning, remediation, and practice. IAPPGA also provides instructors a teaching tool that can be used in or out of the classroom for graph theory and data structures, and other computer science courses that are related to graph algorithms. IAPPGA includes a set of Java applets and html files so that the user can work online.

2.1 Design of IAPPGA

IAPPGA organizes the basic graph algorithms in different categories based on the feature of the graph: undirected

graphs and directed graphs. Each category splits into un-weighted graphs and weighted graphs for fast accessibility. The algorithms that apply to each type of graph are assembled together. Users may simply select the right type of program to avoid extra input information required by other types of graphs, such as weights. When I introduced this package to faculty peers, there were different comments on it – some instructors would like to see a more integrated format, while some instructors would like a more straight and simple way to get into the type of graph needed. Making the link simple is actually a good idea for fast Internet access.

2.2 Input format

IAPPGA allows users to input a graph either by online drawing (this works well for small graphs) or by entering the adjacency matrix (simply type the adjacency matrix in, or copy and paste the adjacency matrix of a graph into the required area). The matrix input allows a free format, i.e., a user may input the adjacency matrix in a single row or break the entries into several rows as long as the entries are in the order of top row to bottom row and from left to right of the adjacency matrix. This package detects any input errors once the user has submitted the adjacency matrix, and provides an error message via a display box to avoid unpredicted wrong solutions. IAPPGA also controls the possible entering of illegal vertex label(s) that are provided by a user online. For example, while working on the flow network, the default source and sink are the first vertex and the last vertex. However, the user may change the default setting so that any vertex can be the source or sink. If a user provided an illegal vertex label, the program will provide an error message and prompt the user to change it.

2.3 Graphical display

When a user inputs a graph by its adjacency matrix, IAPPGA will draw the graph on a graphical canvas. Of course, the program draws the graph following a "predetermined rule" and it might not appear as the shape that the user would like to see. On each working page of this package the user may drag any vertex of the graph to another location so that the display of the graph appears as the user wants. Even for online drawing, the user may change the shape of the graph while running a specific program to obtain the best view. IAPPGA can maintain the graphical display the user adjusted from one algorithm to another.

2.4 The features of IAPPGA

This program package does not require downloading or installing any software. A computer with Internet accessibility is enough to run all the programs. The package has the following features:

(a) Online user interface for easy use. There are online instructions to help a user utilize the online drawing feature. An additional page was designed to introduce the adjacency matrix to users who are not familiar with graph theory.

(b) Preview of key ideas and steps for each algorithm online. Students can get quick help on the key idea of each algorithm in order to better understand the process.

(c) Ability to solve practical graph theory problems online. Online drawing allows a user to draw a graph with up to 25 vertices.

(d) Ability to see and study the solution output that provides the details of the working process step-by-step for each algorithm.

(e) A color and adjustable graphical display to visualize the solution on the actual graph, so the user will have a better understanding. The adjustability allows a user to change the shape of the graph to get the best view of the graph and its solution.

(f) Allows the user to input the information of a graph by online drawing or using an adjacency matrix – the most popular input method. The online drawing works well for small graphs; the adjacency matrix method provides a fast input method.

2.5 An example of IAPPGA

The Edmonds-Karp Max-flow & Min-Cut algorithm is one algorithm of IAPPGA and is a popular example for visualization of a graph algorithm. The following pictures show how IAPPGA works on this algorithm. First, assume that the user selected matrix input.

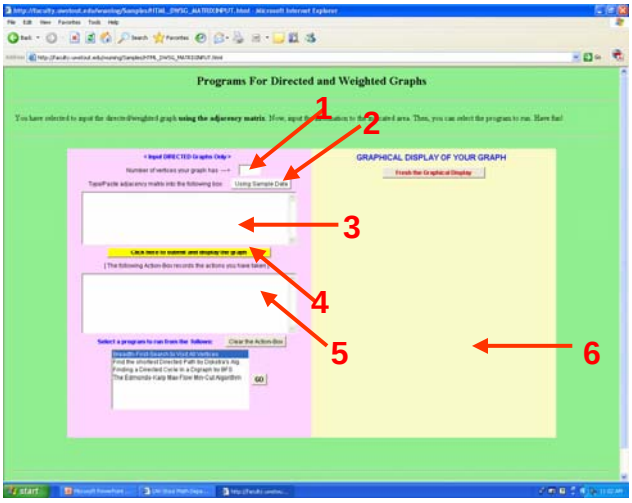

Figure 1: Home page if selecting matrix input for a weighted digraph.

On Figure 1: The user will input the number of vertices of the graph in text box (1) and its adjacency matrix into text area (3), then click on button (4) to submit. After submitting the data, the user will see the graphical display on canvas (6) and the matrix submitted in text area (5). Error messages also appear in (5) if user inputs invalid information. For users who are not quite familiar with an

adjacency matrix, either they can learn about the djacency matrix online (from the home page of IAPPGA) or click on button (2) to get a sample graph with seven vertices and its data appearing in box (1) and text area (3).

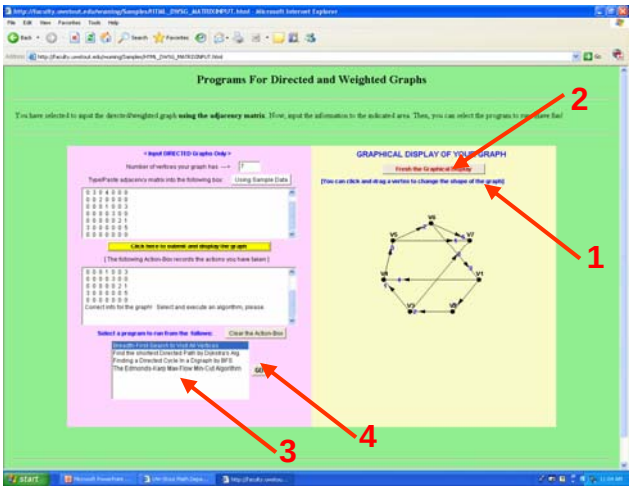

Figure 2: Display after clicking on "Using sample data" button (2) and then submit button (4) on Figure 1

On Figure 2: A sentence (1) prompts the user to change the shape of the graph if a better view is desired. The user may select a program from list box (3) and click on button (4) to run a practical program. Notice that the graphical display might be interrupted due to switching layout pages back and forth. A user may click on button (2) any time to refresh the graphical display.

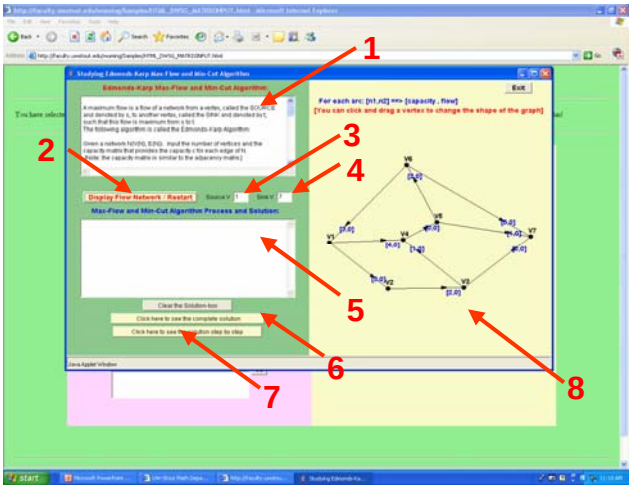

Figure 3: Executing Edmonds-Karp Max-Flow & Min-Cut algorithm [after selecting the program and clicking button (4) on Figure 2]

On Figure 3: Text area (1) reviews the key idea and steps of the Edmonds-Karp algorithm. Clicking on button (2) will show the graph on canvas (8). Here, assume that the user has changed the graph to the shape as shown on the canvas. This program allows a user to apply Edmonds-Karp Max-Flow & Min-cut algorithm with an arbitrary source and sink. The default source and sink are 1 and n if the network has n vertices. However, the user may change the source and sink via input boxes (3) and (4). Button (6) gives the fast solution and button (7) allows the user to see the solution process step by step. Whichever of buttons (6) or (7) was clicked on, the detailed description of the solution will be displayed in the solution box (5) and on the graphical display (8).

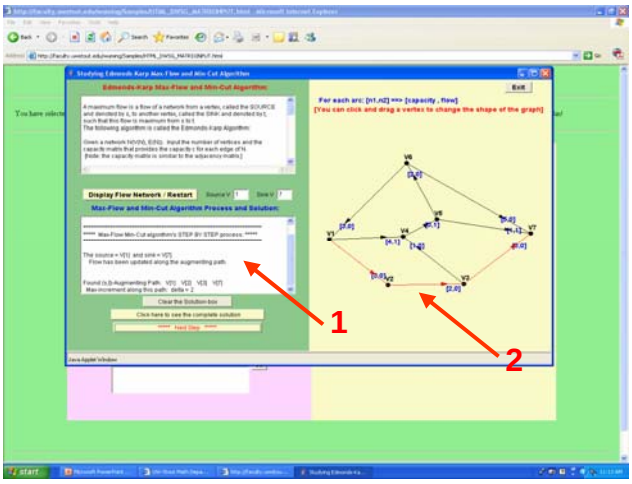

Figure 4: Step-by-Step solution for finding the maximum flow and minimum cut

On Figure 4: This page shows the step-by-step solution if the user clicked on button (7) of Figure 3. After the first clicking on this button, the label of this button becomes "Next Step" to prompt the user to keep going until the process is completed. Repeatedly clicking on the same button, the working process will be displayed simultaneously in the solution box (1) and on the graphical display (2) step-by-step. Figure 4 shows the stage that the first augmenting path from source to sink was found: the path is indicated in red color on the graphical display (2) and stated in solution box (1). The solution box indicates that the network flow along this path may be increased by 2 units. The user can check the data on the graphical display to compare the capacity and flow on each edge to verify the progress. The program will repeatedly find the augmenting path until there are no more such paths.

The final solution including the maximum flow and the Min-Cut subsets will be displayed in the text area (1) when it is all done as shown in Figure 5 on next page.

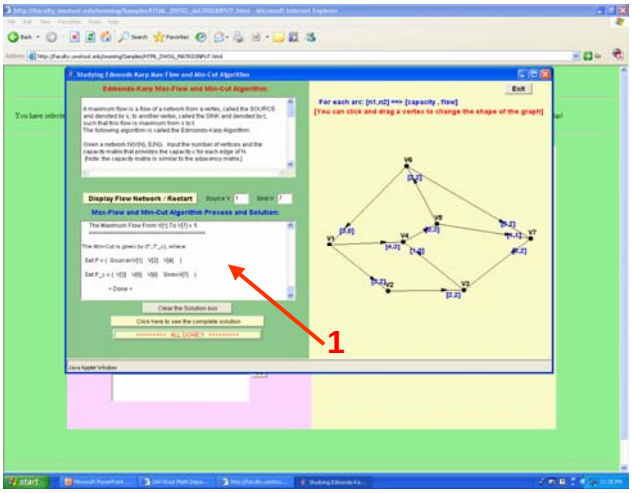

Figure 5: Final solution of Edmonds-Karp algorithm for finding Max-Flow & Min-Cut to this network

By combining the key idea of the algorithm and the text display of the solution along with the graphical display, students should be able to understand the whole process of an algorithm. I have developed this package so that each program will have these features if applicable.

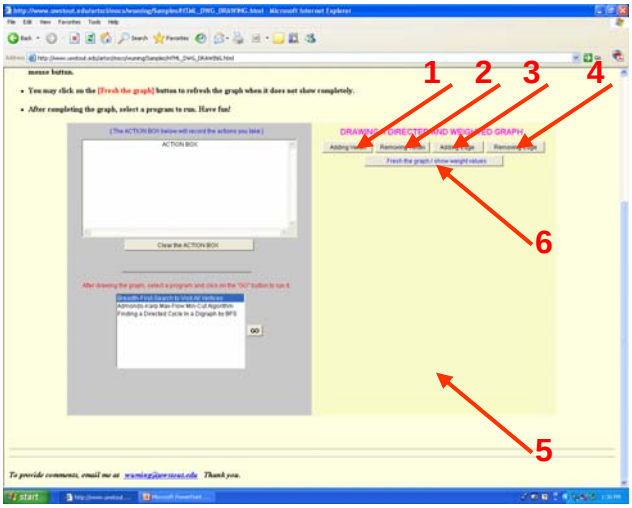

Figure 6: The main page if selecting online drawing a weighed digraph.

On Figure 6: This figure shows the working environment if a user selected online drawing. Online instructions are available for each type of graph. There are four buttons for adding/removing vertices or edges indicated by (1), (2), (3), and (4). Each one of these buttons allows the user to repeatedly add or remove vertices or edges on the canvas (5). A popup window allows user to input the weight value. When the pup-up window interrupts the graphical display, the user may click on button (6) to refresh the graphical display. Once the drawing is completed, the user may select a program to execute.

Each program works the same way as if a matrix was input (introduced earlier).

When selecting online drawing, users would certainly like to draw the graph in a convenient way – but it might not be the best way to view the solution. IAPPGA allows users, while executing an algorithm, to change the shape of the graph s/he drew to receive the best view.

3. Using IAPPGA in the classroom

I have used this package in my Graph Theory class for several semesters including an older version of IAPPGA and the current version. Once I introduced this package in the classroom and demonstrated it via a few examples, students can run these programs at home or in a computer lab without any difficulty. I allowed students to use this package for their written homework as well, e.g., to double check their solution if applicable. Homework programming projects were assigned as two parts: first, using the online program to find the solution for several practical graphs and submit a written report of the result; second, implementing the algorithm on a computer using a Java language program. Students like IAPPGA and its features, and have provided valuable feedback. I fixed quite a few "bugs" based on students' comments. The step-by-step solution makes students think, "What will happen next and why?" All students were able to use this package to review the key steps of algorithms, solve practical graph problems, and gain a better understanding of material.

Since I started using IAPPGA, student's learning and performance has improved significantly. Some specific areas of improvement include:

- With this package, the Instructor does not have to spend a lot of time drawing pictures and explaining the working process of an algorithm. Furthermore, this package allows the instructor to conveniently demonstrate the algorithm repeatedly using different data. This has allowed me to cover about 10% more material than when I was not using the package.
- Students may review the algorithm and its working process via this package at anytime. When working on their written homework assignments, students are able to double check their solution via this package. I found that, if the assignment is algorithm related, most students can get sufficient help from this package instead of stopping by the instructor's office to ask for further explanation. As a measure of the impact, the correctness of homework solutions has increased from 80% to 95%.
- When working on their programming projects, students have a much faster and better understanding. According to my records, with this package, I can assign seven to eight programming projects – two more than without this package.

To my surprise, through using this package, students seemed eager to write their own program using object-oriented technique and making their own programs with GUI features as well, even if it was not required.

4. Conclusion

To teach a graph algorithm, simply providing source code does not make too much sense. The purpose of IAPPGA is to help students have a better understanding via solving a graph problem and investigating the solving procedure. I have studied other educational web sites such as [8], and [9]. These web sites are willing to teach graph theory online. It would be very nice if students could work on websites that teach graph theory concepts along with an easy online package to practice and see the applications as well. IAPPGA is so designed and it is available online. I invite the instructors from different schools to apply IAPPGA in the classroom and encourage students to learn the algorithms via this software.

The author has planned further development of this package. The main goal (based on user comments and resources such as [6]) is to add all basic graph algorithms to IAPPGA. The author has previously written programs for deadlock detection of computer networks, typical sorting and searching techniques. These programs will be added to IAPPGA as well. In order to make this package more user-friendly and more efficient, the author appreciates any comments and suggestions. Feedback may be sent to wuming@uwstout.edu. Programs of IAPPGA are available online at http://faculty.uwstout.edu/wuming/Samples/SelectInputMethod.html.

REFERENCES

[1] Gary Chartrand and Ortrud R. Oellermann, textbook: "Applied and Algorithmic Graph Theory", 1993.

[2] Thomas Cormen, Charles Leiserson, and Ronald Rivest, textbook: "Algorithms", 1994 (12th Print).

[3] Stuart Hansen, Karen Tuinstra, Jason Pisani and Lester I. McCann, Graph Magic: A visual Graph Package for Students, Computer Science Education, Volume 13, Number 1, March 2003, page 53-66.

[4] Jon Harris' JGraphED: http://www.jharris.ca/JGraphEd/JGraphEd.htm

[5] Khuri, S. and Holzapfel, K. (2001), An educational visualization environment for graph algorithms. Proceedings of the 6th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education, pages 101 – 104.

[6] William Kocay and Donald L. Kreher, textbook: "Graphs, Algorithms, and Optimization", 2005.

[7] Douglas B. West, textbook: "Introduction to Graph Theory" (2nd edition), 2001.

[8] Simon's Rock College of Bard, a web site that teaches graph theory concepts online: http://www.simons-rock.edu/~jlegge/thesis/index.php

[9] Other samples of graph programs hosted by universities (web sites):

- Department of computer science, University of Toronto, Canada: http://www.dgp.toronto.edu/people/JamesStewart/270/9798s/Laffra
- Fakultät für Informatik der Technischen Universität München: http://wwwmayr.informatik.tu-muenchen.de/EVEGA/source.html
- Department of Mathematical Sciences, Northern Illinois University, USA: http://www.math.niu.edu/~rusin/known-math/index/05CXX.html#COMP
- Saratov State Technical University, Russia: http://www.geocities.com/pechv_ru