gsearchGUI v.1.0

A Graphical User Interface for

Graph Search

K.Tobakidis, Ath. Kehagias, G. Hollinger and A. Gelastopoulos
10 Oct 2010

Introduction

gsearchGUI provides a graphical interface for several **graph search** algorithms. The graph search problem is a version of pursuit-evasion, taking place on a graph: a team of pursuers try to locate a **mobile** searcher moving inside a graph.

The algorithms used by ***gsearchGUI*** are ***GSST***, ***Extra Nodes*** and ***Extra Searcher***. GSST clears a graph with a *node-located* intruder by an *internal, monotone, connected (IMC)* search; the algorithm is described in "Anytime Guaranteed Search using Spanning Trees" and "Searching the Nodes of Graph: Theory and Algorithms"). *Extra Nodes* and *Extra Searcher* clear a graph with an *edge-located* intruder by an *internal, monotone, connected* search; the algorithms are described in "Pursuit-Evasion Problems using Graph Theory methods: Software development and experiments" (in Greek). The entire GUI package is available at Thanasis Kehagias page. The gsearch.exe package is available at Geoffrey Hollinger's software page.

To Top

Quick Start

(`GsearchGUI` is distributed for use as is, without any explicit or implicit warranty. Feel free to use it and disseminate it.) Download `gsstGUI.zip` and unzip it into a directory of your choice; then run `gsearchGUI.exe`. You will see the following screen.

-

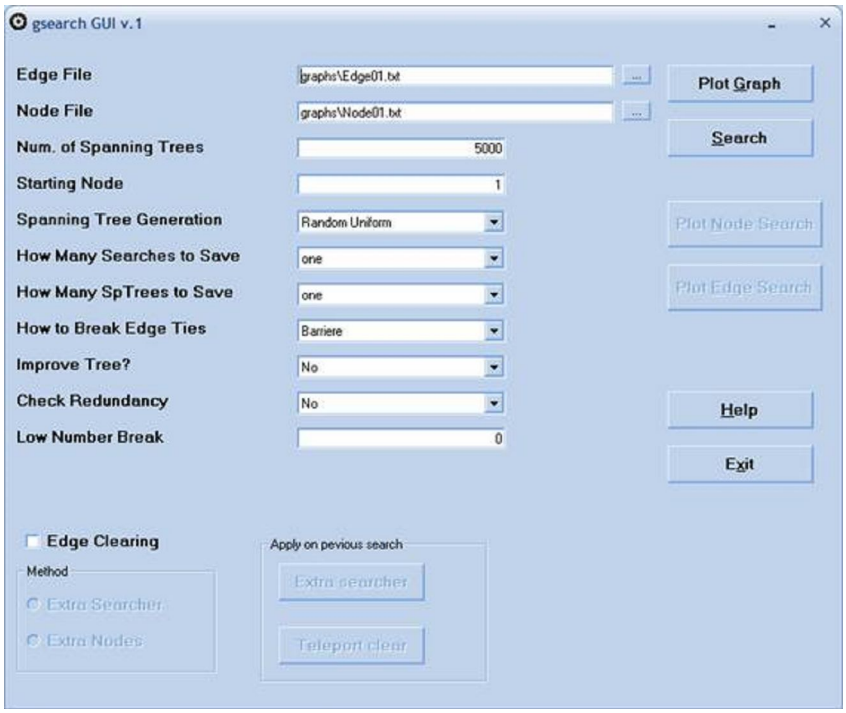

Figure 1. A screenshot of the gsearch GUI.

Create and view the image of the graph by pressing the "Plot Graph" button.

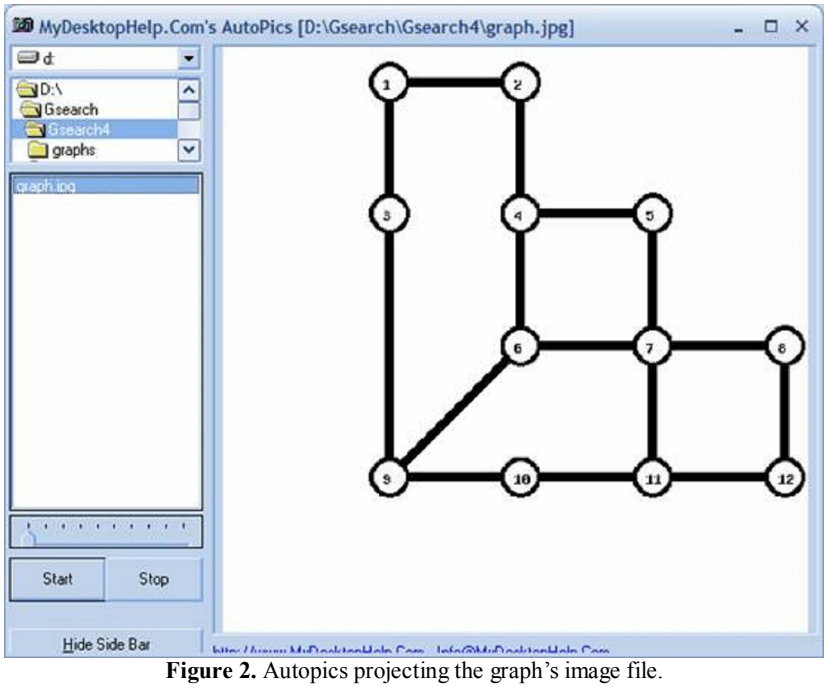

Figure 2. Autopics projecting the graph's image file.

Close the plot window and, in the screen of Figure 1 press the "Search" button. A graph search will be performed and soon you will see the following message.

Figure 3. Autopics projecting the graph's image file.

Press OK to close the message window. Now press the "Plot Node Search". The following screen will appear.

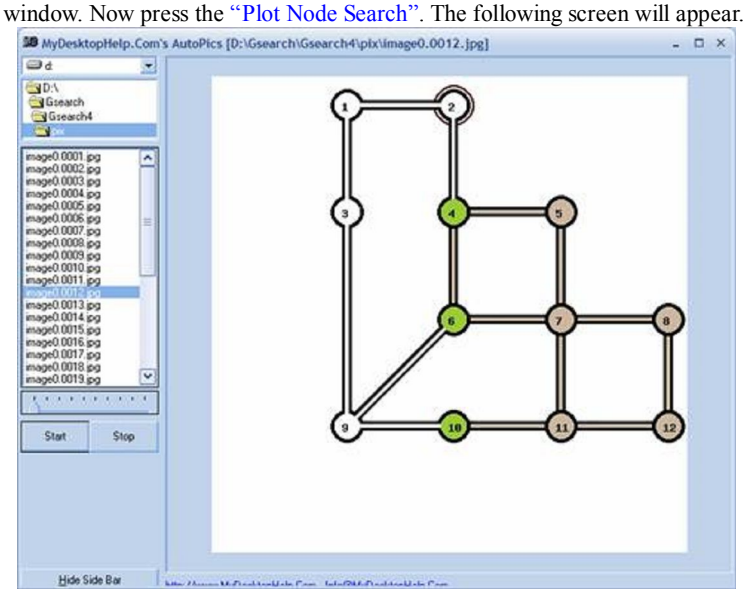

Figure 4. *Autopics* presenting a slide show of the graph search.

Press the Start button to see a slide show of the graph search.

If you get some error message, it is most likely due to operating system (XP, Vista, Win 7) compatibility issues. See Installation for possible solutions.

To Top

Installation

Installation of `gsearchGUI` is extremely simple: download `gsstGUI.zip` and unzip it into a directory of your choice; then run `gsearchGUI.exe`. `GsearchGUI` is distributed for use as is, without any explicit or implicit warranty. Feel free to use it and disseminate it.

When first running the program, if an error message regarding the msstdfmt.dll file is shown, run the *msstdfmXP.bat* file, located in the main folder. This problem mainly occurrs on Windows XP OS. The *msstdfmXP.bat* command must be run with Administrative rights. On Windows Vista or 7 a similar problem can occur, regarding the comdlg32.ocx file. Try running the *Vista_win7_run_as_admin.bat* by Right click-> Run as Administrator.. If the program still does not run properly on Windows Vista or 7, try using the compatibility mode.

To Top

Usage

The program corresponds closely to the command line components, as described in the Components Section. Launching the program brings up the window of Figure 5. On the left side we see several input boxes. All of these correspond to options of the *gsearch.exe* program (for an

explanation of the options see the next section of this document or "Searching the Nodes of Graph: Theory and Algorithms")) with one exception: in addition to the edge list file (`graphs/Edge01.txt` in Figure 1) there is also a file containing the *x*- and *y*-coordinates of the nodes (`graphs/Node01.txt` in Figure 1 -- this is required for visualization). At bottom, there are the options of the functions of the *Eclear.exe* program (as described in the Components Section and "Pursuit-Evasion Problems using Graph Theory methods: Software development and experiments"), using which we can run an edge-clearing search.

-

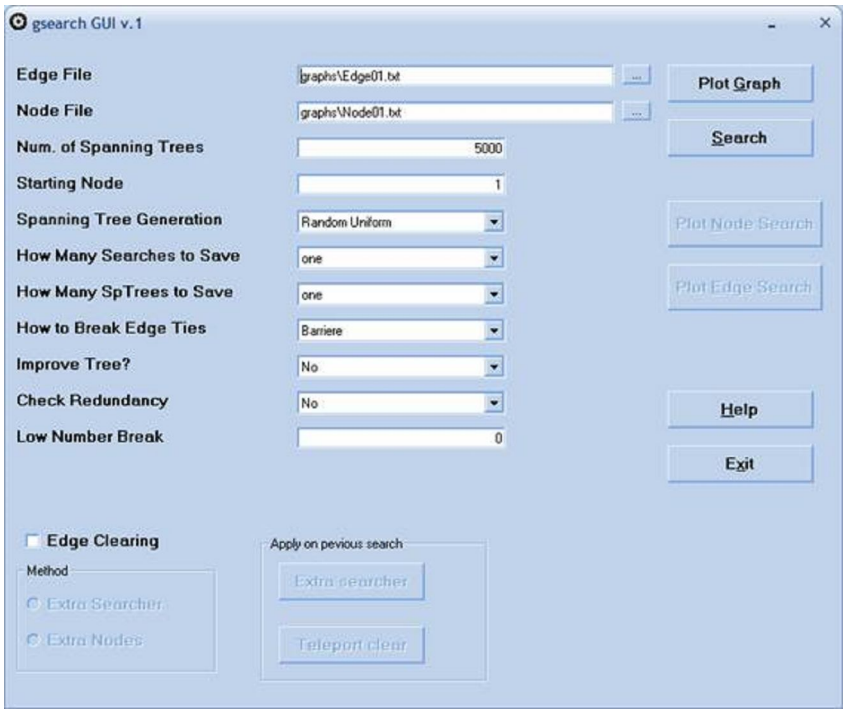

Figure 5. A screenshot of the gsearch GUI.

On the right side there are several command buttons:

1. The "Plot Graph" button creates an image of the given (always named as `graph.jpg`) and invokes the *AutoPics* (described in Components Section) image viewer to project it.
2. The "Search" button performs the graph search (it actually invokes the `gsearch` and if needed the `Eclear` executables and stores their results in the `output` directory). Time to complete the search depends on the size of the graph and the number of spanning trees used.
3. The "Plot Node Search" and "Plot Edge Search" buttons plot the graph search for the node or the edge game. Actually, first the search strategy files are used to generate a sequence of image files, one image corresponding to each step of the search (these files are stored in the directory `pix`) and then the image viewer is invoked to view the files. The user can step through the images using the arrow keys, or use the start / stop buttons to run a slide show of the search.
4. The "Help" button invokes the current file.
5. The "Exit" button terminates the program.

When starting the program, as we see in Figure 5, some buttons are by default disabled. Those buttons cannot be used as there is no search strategy saved yet. Also the "Extra Searcher" and "Extra Nodes" radio button will only be enabled when the "Edge Clearing" checkbox is checked. Let's see an example of the program's usage, following the next steps:

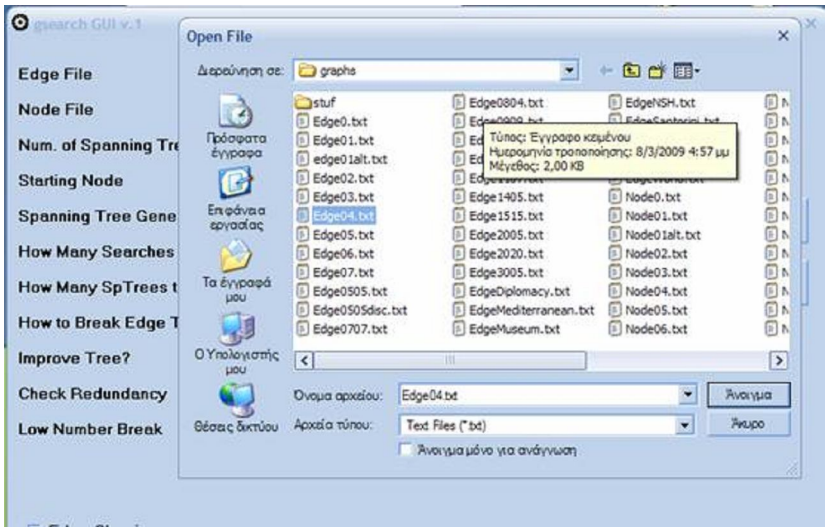

Figure 6. Choosing the graph files.

1. Choose the edge and node files.The GUI starts with the default selection of `Edge01.txt` and `Node01.txt` ; if you want a different

graph, type the corresponding file names in the input boxes or click the browsing buttons (Labelled with "…"). Some examples of these files are located in the `graphs` directory.

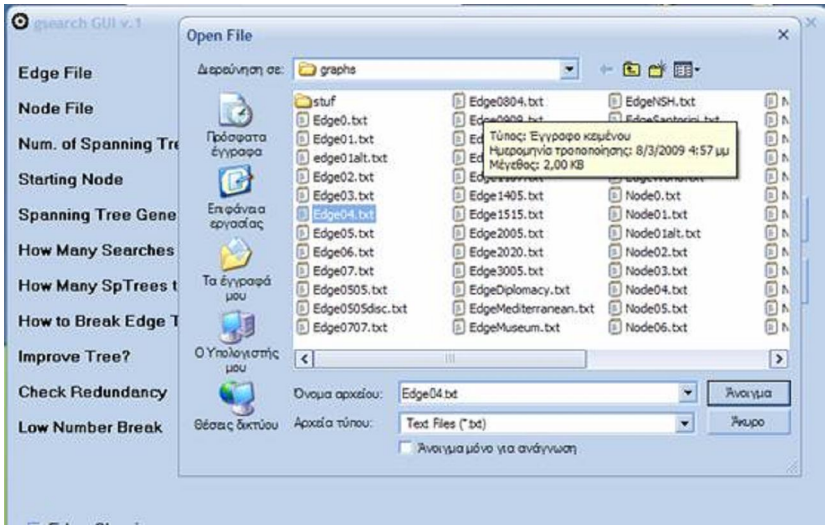

Figure 7. Browsing the example files in the `graphs` directory.

2. Create and view the image of the graph by pressing the "Plot Graph" button.

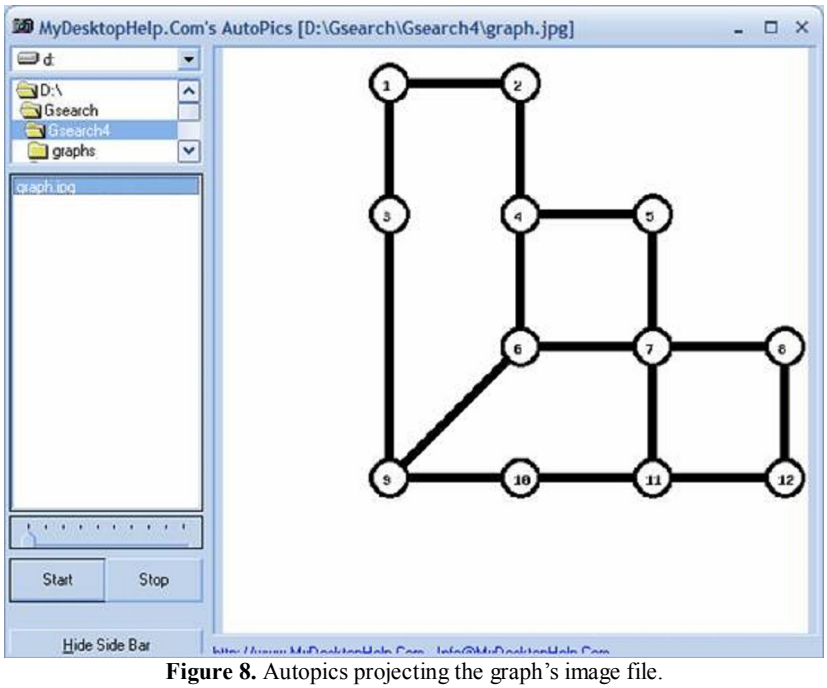

Figure 8. Autopics projecting the graph's image file.

3. Choose the arguments of gsearch from a combobox list.

Figure 9. Choosing how to break edge ties.

4. If we wish to perform an edge-clearing search we check the checkbox bellow and choose the desired algorithm. This step can be omitted if we wish to perform a node search.

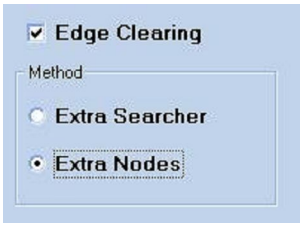

Figure 10. Edge clearing options.

5. Perform the search by pressing the "Search" button. While the necessary commands are running, a small tool called *Gsearch killer* will pop up. In case the search is taking too long, we can interrupt it by pressing the "KILL IT" button. Afterwards we can change our options to minimize the search time and rerun the search.

Figure 11. Gsearch killer tool.

When the search is over, a message box will inform us for the minimum number of searchers we could achieve.

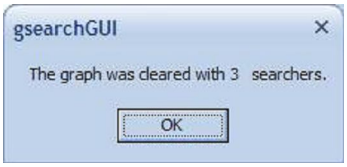

Figure 12. Message box after the search runs.

6. We can view the saved search strategy for the node (edge) game by pressing the "Plot Node (Edge) Search" buttons, which are now enabled. Every node of the graph are colored with light green if one searcher is present, with dark green if more than one searchers are present and with white if a searcher has visited it in the past steps of the search. The edges of the graph are colored with white if they have been node or edge cleared (depending on the game we chose to visualize). Every element of the graph, node or edge, than hasn't yet ever been clear is colored with light brown. The root of the search is highlighted by a brown circle around it. We can follow the search either manually or by using the "Start" and "Stop" buttons.

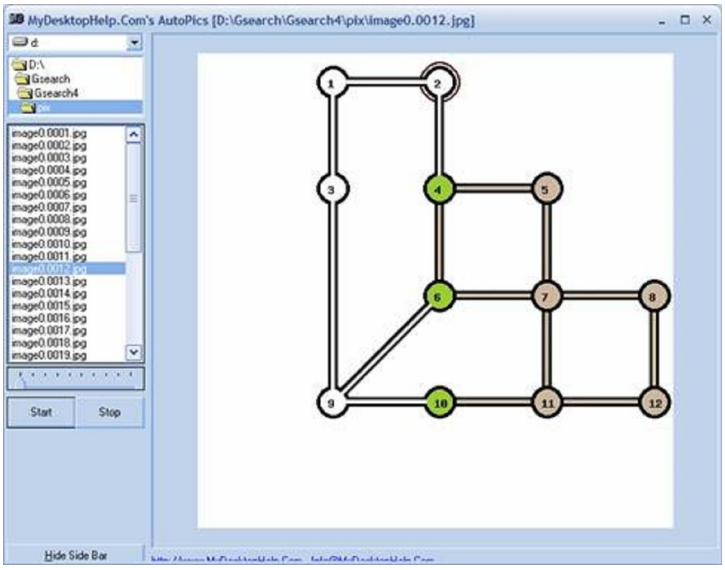

Figure 13. Edge game search visualization. The edge {4-6} would be considered as clean for the node game.

7. If we omitted the fourth step, meaning we performed a node search, we can now apply the "Teleport Clear" or "Extra Searcher" functions on the saved node search. The program informs us when the operation is over.

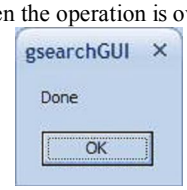

Figure 14. End of teleport clearing or applying the Extra Searcher algorithm on a saved search.

Press "OK" and plot the search once again. Observe the possible differences from the initial search.

8. We can change our options and run as many searches we wish; then exit the program.

To Top

Components

gsearchGUI interfaces the following executable files.

1. The command line program gsearch.exe. The program was implemented by G. Hollinger in ANSI C using the gcc 4.1 compiler. For the sake of completeness we also list the help file of `gsearch.exe`.


```
USAGE:   gsearch -m [graph] -n [no.trees] -s [startnode] -g [gen.tree] -y [wr-search] -w [wr-tree] -v [visualize]
         -t [edge traversal], -i [improve tree] -l [low number], -r [redundancy check]
EXAMPLE: gsearch -m graphs/Edge14.txt -g exhaustive -s 1 -n 500000 -i 0 -t bh -y one -w one
```

The various options available are as follows.

```
-m [graph]: string, name of file with edge list of the graph.
-n [no.trees]: int, how many sp.trees to generate (DEFAULT is 1).
-s [start node]: int, which node to start (DEFAULT is 1, random choice is 0)
-g [gen.tree]: string, method of generating spanning trees
(acceptable values: readtree, exhaustive, uniform, dfsrand; DEFAULT is uniform)
-y [wr-search]: string, how many best searches to write
(acceptable values: one, all, none; DEFAULT is one)
-w [wr-tree]: string, how many best sp.trees to write
(acceptable values: one, all, none; DEFAULT is one)
-t [traversal]: string, how to break edge label ties
(acceptable values: bh, random, bhrand, bhweight, bhdom; DEFAULT is bh).
-i [improve tree]: boolean, use tree improvement technique or not.
(acceptable values are 0 / 1; DEFAULT is 0: do not use)
-r [redundancy check]: boolean, check for redundant trees
(acceptable values are 0 / 1; DEFAULT 1: check for redundancy)
-l [low number]: break if a tree is found with this number of searchers
(DEFAULT is 0: do not break)
-v [visualize]: boolean, use visualizer (only supported on linux)
(DEFAULT is 0: do not use)
-h [help]
```

The `-m` option indicates the file which contains the graph description. This must be an ascii (plain text) file containing a list of the edges of the graph, one edge per line, indicated as a pair of nodes. The nodes must be continuously numbered from 1 to *N* and these numbers are used as labels. *N* is assumed to be the largest number appearing in the edge list (the graph is assumed to be undirected and connected). Examples of edge lists can be found in the `graphs` directory. (Many examples of edge files are stored in the subdirectory `graphs`).

The remaining options of `gsearch` correspond to the description of the algorithm as given in the paper cited above. The `-n` option corresponds to the *M* parameter of GSST (number of spanning trees), `-s` is the root of the search, `-g` describes the uniform and DFS methods for generating spanning trees (there are also options for exhaustive enumeration of all spanning trees and for reading a specific spanning tree from file) and the `-t` option corresponds to the traversal methods BH, UR, BR, BW and BD. The `-i, -r` and `-l` options are self explanatory. The `-y` option writes one or more minimal strategies in the file(s) `output/strat*.txt`; each row of the file corresponds to one step of the strategy and shows the nodes in which the searchers are currently located. The `-w` option writes the rooted tree(s) corresponding to optimal strategies, in the file(s) `output/tree*.txt`; each row of the file shows one parent and her child. The `-v` visualization option only works on Linux computers (for a visualization of the search on Windows computers use the GUI).

2. The command line program Eclear.exe. The implementation of the *Extra Searcher* and *Extra Nodes* algorithms was created by K. Tobakidis in C using the Dev C++ v.4.9.9.2 compiler. The help of the program is listed bellow.

```
Usage:

Eclear -i Edgefile :(Extra Nodes)
   prints the fake-noded Edgefile

Eclear -o Edgefile Searchfile(with fake nodes) :(Extra Nodes)
   prints the edge clearing strategy in the original graph

Eclear -x Edgefile Searchfile :(Extra Searcher)
   applies the Extra Searcher on a node clearing strategy

Eclear -t Edgefile Searchfile :(Node Search)
   prints the node clearing strategy, teleport-cleared
```

The program can also perform a clearing of the teleporting moves on any *IMC* search strategy, with the –t option.

3. The Gsearch killer program. A program to interrupt the above two command line programs when we wish to. The program is shown at Figure 7 (Quick start Section) It is used as VB6 does not support multithreading so the Operating System's multitasking does the job.

4. The Autopics picture viewer. The image viewer is a freeware stand alone program made available by the company *Mydesktophelp* (at *http://www.mydesktophelp.com/development.php*).

The actual GUI was implemented in MS Visual Basic 6.0 by K. Tobakidis and Ath. Kehagias. The program also uses the freeware graphics library called *GD* available at *http://www.libgd.org/Main_Page*.

To Top

Directory Structure and File Formats

The following directory structure is used by the software. Under the installation folder, there exist the following subfolders.

1. **graphs:** This is where the graph edge and vertex files are stored.
   a. The format of edge files is: one line per edge, with two numbers per line (for the respective nodes). The nodes must be numbered consecutively, from 1 until N.

   b. The format of vertex files is: one line per node, with two numbers per line (the x and y coordinates of the node). The node files are only needed for visualization.
2. **help:** This is where the help file is stored, in html format.
3. **images:** You can ignore this directory, but **do not delete it!**
4. **output:** This directory contains text files which contains output of the *gsearch* executable (see *gsearch* documentation for more details).
5. **pix:** This directory contains the image files used by *AutoPics* to visualize the graph search. You can delete the *.jpg files, but **do not delete** *the AutoPics.exe*!

To Top

Known Issues

1. Regarding OS compatibility issues, see the Installation section.
2. Do *not* use the exhaustive spanning tree enumeration option on large graphs because the program may run out of memory and crash.

To Top

