Homework 3

Artificial Intelligence I — CS475/CS505
Deadline: February 15, 11:55pm

February 7, 2007

The purpose of this homework is to learn how to implement AI systems. Before we can do anything more significantly, we start with the implementation of well-known search strategies. For our first implementation, you are asked to implement the two uninformed search strategies: breadth-first and depth-first search. The input is a graph. To help me in checking you programs, please use one of the languages available in our CS lab: C or C++. The constraints that I would like your program to satisfy are:

- Use an integer array named `nodes` (i.e., `int nodes[]=...;` can be used in the declaration) to store the nodes of the graph;
- Use an integer array named `edges` to store the edges of the graph.

Let us assume that `nodes` and `edges` are global variables. You might have to define additional data structures that are needed for the implementation. For example, you will probably need the queue for the BFS or the stack for DFS. Your program should contain the following procedures/functions:

- `int *getSuccessor(int i)` — this returns the set/list of nodes that are successor of the node `i`;
- `int breadth_first_search(int init, int goal)` — this should return 1 if there is a path connecting `init` and `goal`; and 0 otherwise. You should print out the search information in the following format:

  ```
  iteration number: x1
    node to expand:   n1
    successors:       n2 n3 .....
  ```

  If a path is found, print the path of nodes that need to be traveled from `init` to `goal`.
- `int depth_first_search(int init, int goal)` — this should return 1 if there is a path connecting `init` and `goal`; and 0 otherwise. For this procedure, the same information as in `breadth_first_search` should be displayed.
- In your implementation, please avoid repeated nodes/states.

The program should be able to receive inputs specifying the search strategy used to find the solutions (1 or 2), the initial node, and the goal node, either through command line arguments or through the keyboard input. 1 stands for breadth-first search and 2 is for depth-first search.

What you need to submit: Please submit everything in one .tar file. The file should contains:

- Source code of your program;
- Makefile (or specific instructions for compiler if any);

- A few set of test data that you have used in testing your program. You can have it in your code and commented it out; and
- A README file detailing how to compile/run your programs.

NOTE: You can use Java in your implementation. Let me know what version of JDK you use.