Eric Roberts
CS106B

Handout #52
May 27, 2009

Section Handout #8—Graphs

The purpose of this section is to let you work with graph algorithms before you begin working on the Pathfinder assignment. Each of the questions use the minimal definitions of `graphT`, `nodeT`, and `arcT` from today's class. These definitions are included in the file `graph.h` from the assignment, which is reprinted in Handout #51.

1. Coding depth-first search

Write a function

```
bool PathExists(nodeT *n1, nodeT *n2);
```

that returns `true` if there is a path in the graph between the nodes `n1` and `n2`. In this version of the exercise, implement this function by using depth-first search to traverse the graph from `n1`. If you encounter `n2` along the way, then a path exists. In addition, implement `PathExists` without using the `visited` flag in the `nodeT` structure. This restriction means that you will need to use sets to keep track of where you've been.

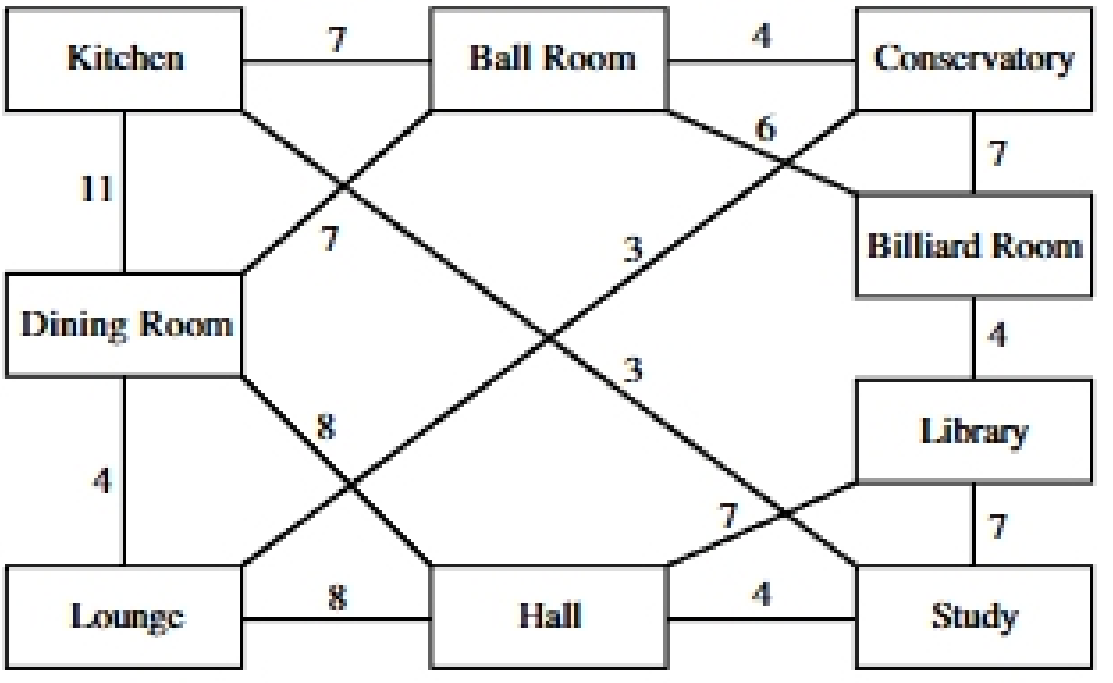
2. Coding breadth-first search

Rewrite the `PathExists` function with two changes. First, change the algorithm from depth-first to breadth-first search. Second, make use of the `visited` flag in the `nodeT` structure to keep track of whether nodes have been visited. You may assume that the client has set all visited flags to `false` initially, presumably by calling a method like this:

```
void ClearVisitedFlags(graphT *gp) {
    Set<nodeT *>::Iterator iter = gp->nodes.iterator();
    while (iter.hasNext()) {
        nodeT *np = iter.next();
        np->visited = false;
    }
}
```

3. Understanding graph algorithms

Those of you who have played Clue will recognize the following undirected graph, which shows the connections between the various rooms on the game board:



The numbers on the various arcs show the distance (measured in spaces on the board) between pairs of rooms. For example, the distance from the Hall to the Lounge is 4 steps, and the distance from the Ball Room to the Billiard Room is 6 steps. In this problem, the "secret passages" that connect the rooms at the corners of the board (the Kitchen-Study and Lounge-Conservatory arcs) are arbitrarily assumed to have distance 3.

3a) Indicate the order of traversal for a depth-first search starting at the Lounge. Assume that iteration over a set chooses nodes in alphabetical order. Thus, the first step in the depth-first search will be to the Conservatory, rather than to the Dining Room or Hall, which come later in the alphabet.

3b) Indicate the order of traversal for a breadth-first search starting at the Kitchen. As before, assume that nodes in any set are processed in alphabetical order.

3c) Trace the operation of Dijkstra's algorithm to find the minimum path from the Lounge to the Library.

3d) Trace the operation of Kruskal's algorithm to find the minimum spanning tree for the Clue board.