CSC 8301 Project # 2

Due Dates: This project has two phases.

- Phase 1: April 5, 2004
- Phase 2: April 19, 2004

Objective: To gain experience in implementing graph algorithms, an understanding of graph properties such as connectivity and acyclicity, and insight in analysis, presentation of results, and peer evaluation.

Background: Sections 1.4 (background on graphs) and 5.2 (graph traversals).

This project has two phases: an implementation and analysis phase, and a peer evaluation phase.

Phase 1: Implementation and Analysis

This part of the project involves implementing a couple of graph search algorithms for finding cycles in graphs and generating random graphs of different sizes and then experimenting with these to answer some interesting questions about cycles in graphs.

- Programs: Each student will implement the following algorithms according to their assigned group. The implementation language is again up to you, but, please, nothing too exotic, since other students in the class will be expected to read and understand your program. Needless to say, clear documentation is also very crucial in this respect.

  (All) The Depth-first Search algorithm using adjacency matrices to determine if a graph has a cycle. The algorithm should stop when a cycle is encountered or when all vertices have been visited and output:
  - the graph
  - the total number of vertices visited
  - whether or not there is a cycle, and,
  - if there is a cycle, the edges comprising the cycle.

  (Group A) The Depth-first Search algorithm using adjacency linked lists to determine if a graph has a cycle. Output same as above.

  (Group B) The Breadth-first Search algorithm using adjacency matrices to determine if a graph has a cycle. Output same as above.

  (All) A Random graph generator that given a probability $p$ ($0 \leq p \leq 1$) creates a graph in which for each pair of vertices there is a probability $p$ that there is an edge between them. Note: This is an undirected graph, so its adjacency matrix must be symmetric and have zeroes all down the diagonal. Random graphs for input to Group A programs using adjacency linked lists should be generated in a similar manner.

- Testing: Test your program extensively on a wide variety of graphs: small, large, sparse, dense to make sure that they correctly identify cycles. Be sure that you test on graphs that are not connected and where the cycle does not occur in the first connected component. Include printouts of some of this testing in your report. To make it easy to see what is going on, draw the corresponding input graph (it is ok to draw this by hand).

- Experiment with your programs and collect statistics.
Group A  Check graphs of different sizes \( n = |V| = 5, 10, 50, 100, 200 \) and different densities (experiment with different values of \( p \), from 0 to 1). For each \( n \) and \( p \) you should run a loop of 100 trials to determine the likelihood that the randomly generated graph has a cycle and how many vertices, on average, where visited. Organize your data in a table. How large does \( p \) have to be so that the probability that a graph with 100 vertices has a cycle is at least 0.5? How large does it have to be so that the probability becomes at least 0.99? Do these answers remain the same as the number of vertices increases/decreases?

Group B “Race” depth-first and breadth-first search on graphs of different densities for \( n = 200 \) to find a cycle:
- Which one visits fewer nodes?
- Which one is faster? (Time them)
- Do you verify a linear correlation between the above?

● Report: Submit four copies of the following:
1. A cover sheet with your name, class section, group, and contact information.
2. Introduction—explain the purpose of the assignment (you can borrow from this handout) and how you carried it out.
3. Program listing.
4. Sample program runs (there should be enough to convince anybody that the program works well).
5. Results and discussion—see below.
6. Either:
   (a) Disks with source code and executable; or
   (b) clear directions detailing where the program can be found on the Villanova computers. Specify which system and give path, file name, and instructions for running the program. Make sure that the files are readable.

Present your programs, data, and experimental results in a clear, convincing, and complete manner (remember the 3 c’s). The presentation of the programs should leave no doubt that they work well by showing sample runs. The presentation of the experimental results should be clear with data arranged in tables and a plot to support your conclusions. Summarize the results and discuss any further observations that you made or other related questions that occur to you that might be answered through a similar investigation.

Phase 2: Peer Evaluation

Each project will be evaluated by three other students in the class. Group A students will be evaluating Group B projects and vice versa. You will be supplied with the three projects, grade sheets (much like the one used to grade your first project) and guidelines on how to evaluate the projects. You must:

1. Fill out a grade sheet for each project and assign it a grade. Be critical, but also try to make your comments constructive. Do not neglect to mention the strong points of the work.
2. Write a brief report \((1/2 - 2 \text{ pages})\) with your impressions. What did you learn from the projects that you evaluated? In retrospect, is there something that you would now do differently in your own implementation? How do the projects that you evaluated compare with one another? How did you like this project as a whole?
Your grade for this project will be computed as follows:

80% Own work. Grades given by other students will be taken into consideration, but the final grade for this part will be assigned by me (using the same criteria).

20% Peer evaluation. Your critical work will be graded according to:

- Your understanding of other students’ work.
- Accruteness of your criticism.
- Thoroughness of the evaluation.