Lab 3: Graph Algorithms

Algorithmic Programming Contests 2023-2024

Wietze Koops

October 21, 2024

1 Algorithms

Here is a brief overview of the most important Graph Algorithms.

- Breadth First Search (BFS), using a queue. [$O(V + E)$]
- Depth First Search (DFS), using a stack. [$O(V + E)$]
- Cycle detection can be done using DFS. [$O(V + E)$]
- Floodfill algorithm (a form of BFS). [$O(V + E)$]
- Finding connected components (in an undirected graph) can be done using Floodfill.
- Determining whether a graph is bipartite, and finding the partition (A, B). [$O(V + E)$]
- Dijkstra's shortest path algorithm. [$O(E \log V)$ using a priority queue]

  Dijkstra's algorithm also works for different distance measures, as long as they have the property that adding an edge to a route does not decrease the length of the route.
- Topological sort. [$O(V + E)$]
- Minimum spanning tree (Prim's or Kruskal's algorithm). [$O(V + E)$]
- Bellman-Ford algorithm. [$O(VE)$]

  (single-source shortest path problem allowing for negative edge weights).
- Floyd-Warshall algorithm (all-pairs shortest path algorithm). [$O(V^3)$]
- Tarjan's strongly connected components algorithm. [$O(V + E)$]

2 Reading from the book

Reading from the Competitive Programmer's Handbook by Antti Laaksonen:

- Part II: Graph Algorithms.

  The lecture covered most of Ch.11 (Basics of graphs), Ch.12 (Graph traversal), Ch.13 (Shortest paths), Ch.15 (Spanning trees) and Ch.16 (Directed graphs).

  Of course, the other chapters are also useful, but won't be covered in this lecture/lab.

3 Lab problems

The problems are from the CSES problem set and from Kattis. The problems CSES are generally focused on implementing a standard algorithm, although some of the later listed ones are more difficult. They are sorted here to ensure increasing difficulty and a mix of topics. The problems from Kattis are from old contests, and therefore generally require more creativity.

- (CSES problem set) Message Route • Hint: 10
- (CSES problem set) Shortest Routes I • Hint: 7
- (CSES problem set) Building Teams • Hint: 16
- (CSES problem set) Course Schedule • Hint: 9
- (CSES problem set) Counting Rooms • Hint: 15
- (CSES problem set) Labyrinth • Hint: 10
- (BAPC preliminaries 2018) Jurassic Jigsaw • Hints: 17 · 4
- (BAPC preliminaries 2017) Crowd Control • Hints: 7 · 14
- (BAPC 2017) Detour • Hints: 7 · 6
- (CSES problem set) Building Roads • Hints: 15 · 3
- (CSES problem set) Monsters • Hints: 10 · 1 · 12
- (BAPC preliminaries 2018) Bee Problem • Hint: 15
- (BAPC 2020) Elevator Pitch • Hints: 15 · 5
- (BAPC 2020) Generators • Hints: 17 · 11
- (CSES problem set) Round Trip • Hint: 13
- (CSES problem set) Round Trip II • Hint: 13
- (CSES problem set) Shortest Routes II • Hint: 8
- (CSES problem set) High Score • Hint: 2

4 Assignment

4.1 Assignment instruction

The lab assignment is due two weeks after the lab session. If you cannot meet this deadline due to exceptional circumstances, send an email before the deadline to `apc@rug.nl`.

Deliverable, to be sent to `apc@rug.nl` before the deadline:

- For each problem that you solved, your solution and a one or two sentence summary of your solution. In addition, you can send in any failed attempts if you are unsure why they failed to receive feedback on them.
- For each problem that you did not solve, your failed attempts. If you already know/suspect the reason why it doesn't work, also mention this.

  If you didn't write any code (because you had no idea where to start), write a few paragraphs on what you tried; in particular, did you figure out why the answer to the Sample Input is what it is? Did you think about the required complexity? What techniques/data structures/algorithms did you consider?

Please include the word 'assignment' in the subject and body of your email.

You may discuss solution strategies with other students. However, the solution that you submit via email (Deliverable) must be written completely by yourself.

4.2 Assignment problems

The assignment problems corresponding to this lab are:

- (CSES problem set) Flight Discount
- (CSES problem set) Road Reparation
- (NWERC 2019) Firetrucks Are Red

5 Hints

1. We can determine for each cell the distance from a nearest monster to that cell by running a BFS starting with all cells with monsters in the queue.
2. Use the Bellman-Ford algorithm.
3. Each added road decreases the number of connected components by at most 1.
4. Construct the fully connected graph on the strings, where the edge weight is the unlikeliness, i.e. the number of positions in which the strings differ.
5. Add the elevators iteratively. Each time, there must be an elevator reaching the highest point not yet reachable. Determine which other cells become accessible using that elevator.
6. Use Dijkstra's algorithm to compute the shortest routes from Amsterdam to every city. Since the graph is undirected, this also provides the shortest routes from each city to Amsterdam, and in particular the first edge of each of these routes.
7. Use Dijkstra's Algorithm.
8. Use the Floyd-Warshall algorithm.
9. Use Topological Sort.
10. Use Breadth First Search (BFS).
11. Construct a graph by adding a central power node, and use this to model the construction cost of power plants.
12. Run a BFS from the start node, where we only add a cell to the BFS queue if its distance to the start is smaller than the distance to the nearest monster as computed earlier.
13. Use Depth First Search (DFS).
14. We can also use Dijkstra's algorithm to maximize the minimal edge capacity, for example by first multiplying all edge capacities by -1 and then minimizing the maximum.
15. Calculate the connected components. Recall that this can be done by starting a BFS from each vertex that was not already visited by an earlier BFS.
16. Determine whether the graph whose vertices are the pupils and whose edges are the friendships, is bipartite.
17. Compute a minimum spanning tree.