
Quiz 2 Review

Contents

1	Sorting	2
1.1	Radix Sort	2
2	Unweighted Graphs	3
2.1	General Information	3
2.2	Searching	3
2.2.1	General	3
2.2.2	Breadth First Search (BFS)	3
2.2.3	Depth First Search	4
2.2.4	Topological sort	4
3	Weighted Graphs	5
3.1	Weight Functions	5
3.2	Single-Source shortest paths	6
3.2.1	General structure	6
3.2.2	Bellman-Ford	7
3.2.3	Dijkstra's Algorithm	8
3.2.4	DAG Shortest Paths	9
4	Dynamic Programming	10
5	Advanced Topics	10
5.1	Bucket Sort	10
5.2	2-way searching	10
5.3	Single-Source Single-Target	10

Although Quiz 2 will focus on material learned since Quiz 1, no material is off-limits. This review handout covers the material taught between Quiz 1 and Quiz 2.

1 Sorting

1.1 Radix Sort

1. Describe the sorting algorithm at a high level.
2. Write out the algorithm in psuedo-code.
3. What is the running time of radix sort? How does the choice of radix affect the running time?
4. What benefits (pitfalls) does this algorithm have?
5. Is this sort stable? In place?
6. What are the requirements for the subsort used?

2 Unweighted Graphs

2.1 General Information

1. Define the range of $|E|$ in terms of $|V|$ given the following constraints on the graph G :
 - No constraints
 - Connected graph
 - Sparse graph (this isn't well-defined, but roughly speaking)
 - Dense graph (this isn't well-defined, but roughly speaking)
2. What does it mean for vertex u to be reachable from v ?

2.2 Searching

2.2.1 General

1. Provide pseudocode outlining the general structure of graph search algorithms.

2.2.2 Breadth First Search (BFS)

1. How does the queue work for BFS?
2. What is the running time of BFS?
3. Does BFS find shortest paths? (measured in # of nodes)
4. What happens if there is a tie in the queue?
5. What are the space requirements?
6. Does BFS traverse the entire graph? How would you alter BFS to guarantee that it visits every vertex?

2.2.3 Depth First Search

1. How does the queue work for DFS?
2. What is the running time of this algorithm?
3. Does DFS find shortest paths? (measured in #nodes)
4. What happens if there is a tie in the queue?
5. What are the space requirements of DFS?
6. Does DFS traverse the entire graph?
7. Describe the following edge types generated by DFS and their significance:
 - Tree edges
 - Forward edges
 - Back edges
 - Cross edges

2.2.4 Topological sort

1. What is a topological sort?
2. What constraints does the algorithm place on its input graphs?
3. Describe a topological sorting algorithm
4. Does a graph have a unique topological sort?

3.2.2 Bellman-Ford

1. What constraints does Bellman-Ford place on its input graphs?
2. Describe the algorithm at a high level.
3. Write out the algorithm in psuedo-code.
4. What is the running time of Bellman-Ford? (best case? worst case?)
5. What benefits (pitfalls) does this algorithm have?
6. Does the ordering of the edge traversal matter? Why or why not? Could traversal orderings change between rounds?
7. How does Bellman Ford handle unreachable negative weight cycles? Reachable ones?
8. What are its space requirements?

3.2.3 Dijkstra's Algorithm

1. What constraints does the algorithm place on its input graphs?
2. Describe at a high level the algorithm
3. Write out the algorithm in psuedo-code
4. Runtime (best case? worst case?)
5. What benefits (pitfalls) does this algorithm have?
6. What are the space requirements of Bellman-Ford?

3.2.4 DAG Shortest Paths

1. What constraints does this algorithm place on its input graphs?
2. Describe at a high level the algorithm
3. Write out the algorithm in psuedo-code
4. Runtime (best case? worst case?)
5. What benefits (pitfalls) does this algorithm have?
6. What are its space requirements?

4 Dynamic Programming

1. When would you use dynamic programming? What types of problems does it help solve?
2. What is optimal substructure?
3. What are overlapping subproblems?

5 Advanced Topics

5.1 Bucket Sort

1. How does bucket sort work?
2. What assumption does it make about its input?

5.2 2-way searching

1. Why would you search a graph from both sides?
2. What search algorithms can you use this technique with?

5.3 Single-Source Single-Target

1. Can you solve the Single-Source Single Target problem asymptotically faster than the Single-Source Shortest Paths problem?
2. How do Single-Source Single Target algorithms improve on the running time of Dijkstra's algorithm?