Computer Science 385
Design and Analysis of Algorithms
Siena College
Spring 2018

Lab 10: Greedy Algorithms and Heaps

Due: Start of your next lab session
(that's in 2 weeks, but hopefully you'll finish during lab today)

You will be assigned a partner to work with on this lab. Only one submission per group is needed.

Group members: ____

Learning goals:

1. To gain better understanding of Dijkstra's algorithm for single-source shortest paths in graphs
2. To relate Dijkstra's algorithm to Prim's algorithm to compute minimum cost spanning trees of graphs
3. To learn Kruskal's algorithm to compute minimum cost spanning trees of graphs
4. To review and expand understanding of the heap data structure and the heapsort algorithm

---

Submitting

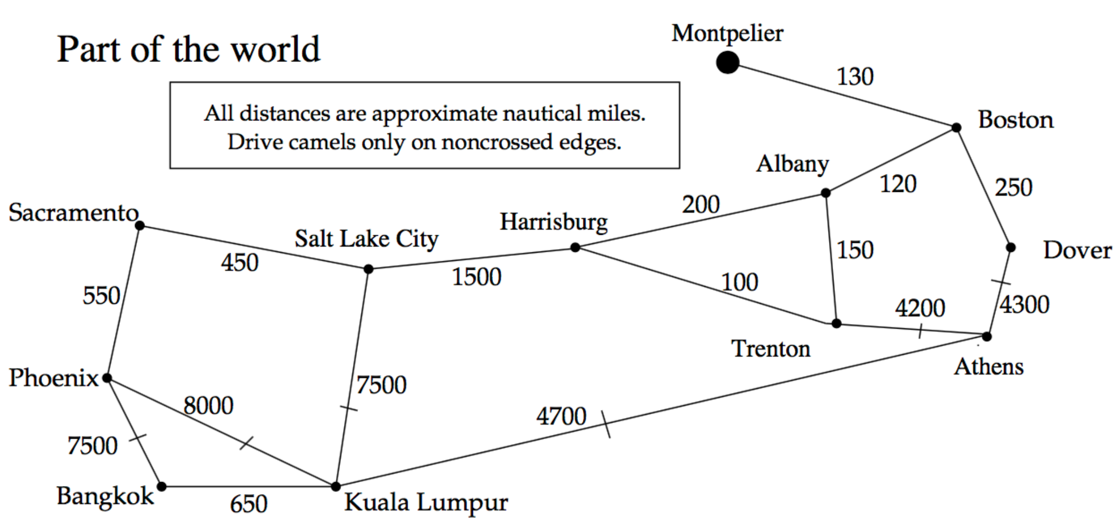
Once all written items are initialed to indicate completion, turn in one copy of this handout. Be sure names of all group members are clearly on the first page.

Grading

Score: ☐ / 100

You are familiar with Dijkstra's algorithm from Discrete Math. Here, the idea is the same: we are looking to solve the single-source shortest paths problem. We will begin with a brief look at the algorithm in METAL and using a small example that we can work through by hand. Once that is complete, you will be ready to answer the questions here about Dijsktra's algorithm.

Question 1:
Using the graph below, (credit: Bailey Problem 16.7, p. 436), use Dijkstra's algorithm to compute the shortest distance from Dover to Phoenix by filling in the tables on the last two pages of this packet (detach them and we can staple them back later), using the algorithm and notation as shown in the example from the start of the lab. (20 points)



Fill in the table in Table 1 which is a map of cities as keys to the shortest distance from Dover to the last edge traversed on that shortest route as values.

It is easiest to specify edges by the labels of their endpoints rather than the edge label itself, which might not be unique.

Also, use the table in Table 2 to keep track of your priority queue. Remember, don't erase entries when you remove them from the queue, just cross them out and mark them with a number in the "Seq" column of the table entry to indicate the sequence in which the values were removed from the queue.

? Question 2:
We studied Prim's algorithm to compute minimum cost spanning trees in class before the break. What change would you need to make to Dijkstra's algorithm to turn it into Prim's algorithm? (3 points)

? Question 3:
In Dijkstra's algorithm, edges are added to the priority queue each time a shortest path is found to a new vertex. The priority values used by the priority queue are computed by adding the length of the shortest path to that new vertex to the length of each outgoing edge that is being added to the priority queue. Describe what you would compute if you were to add 1 to the shortest path instead of the edge length, assuming that the priority queue breaks ties in priority by removing the edge that has been in the queue the longest. (3 points)

? Question 4:
You have seen an algorithm before that computes this. What is it? (3 points)

? Question 5:
How does the efficiency of the modified Dijsktra's algorithm compare to the direct implementation of algorithm you named in the previous question? (3 points)

Read Levitin Section 9.2 about Kruskal's algorithm, and then complete the next two problems.

? Question 6:
Using the style as shown in Levitin Figure 9.5, p. 326, demonstrate the steps of Kruskal's algorithm on the graphs in Levitin Exercise 9.2.1, p. 331, part a. Show your work (5 points)

Question 7:

Using the style as shown in Levitin Figure 9.5, p. 326, demonstrate the steps of Kruskal's algorithm on the graphs in Levitin Exercise 9.2.1, p. 332, part b. Show your work. (8 points)

For the remainder of this lab, you will review and expand your understanding of the heap data structure and the heapsort algorithm. If you have not studied or do not recall the details of heaps and heapsort, please review Levitin Section 6.4 and/or the topic notes on Heaps linked from the schedule page.

Question 8:
Consider the construction of a min-heap by starting with an empty heap, then inserting values in a given order. Show the min-heap after each of the values 2, 8, 97, 31, 12, 3, and 39 are added to an initially empty heap. (6 points)

? Question 9:

Consider the construction of a min-heap as done in the first phase of a heapsort. Start with an array containing the values 2, 8, 97, 31, 12, 3, and 39. Show the contents of the array (in array or tree form, your choice) after each non-trivial iteration of the first phase of the heapsort (which goes from an unsorted array to a min-heap). (7 points)

Question 10:
Finally, show the transformation from the min-heap you just constructed to a sorted array, completing the heapsort procedure. (7 points)

Question 11:
Consider a binary min-heap like the ones in Levitin Section 6.4. Which locations in a binary min-heap of n elements could possibly contain the third-smallest element? (3 points)

Question 12:
Which locations in a binary min-heap of n elements could possibly contain the largest element? (2 points)

The heaps we have seen here, where the heap is represented by a binary tree stored in an array, are one specific case of a more general structure called a *d-heap*. In a d-heap, each node has up to d children. So the binary heaps you would have seen before would be 2-heaps. For the questions below, assume that the minimum value is stored at the root node (*i.e.*, that it is a min-heap).

Question 13:
Show the construction of a 2-heap that results from the following values being inserted: 18, 9, 23, 17, 1, 43, 65, 12. How many comparisons are needed? (8 points)

Question 14:
Show the construction of a 3-heap that results from the following values being inserted: 18, 9, 23, 17, 1, 43, 65, 12. How many comparisons are needed? (8 points)

Question 15:
For the heap element at position i in the underlying array of a 3-heap, what are the positions of its immediate chidren and its parent? (Give formulas in terms of i.) (2 points)

Question 16:
For the heap element at position i in the underlying array of a d-heap, what are the positions of its immediate chidren and its parent? (Give formulas in terms of i and d.) (2 points)

Question 17:

Show the construction of a 1-heap that results from the following values being inserted: 18, 9, 23, 17, 1, 43, 65, 12. How many comparisons are needed? (5 points)

Question 18:

Show the construction of a 7-heap that results from the following values being inserted: 18, 9, 23, 17, 1, 43, 65, 12. How many comparisons are needed? (5 points)

City	(distance,last-edge)
Dover	(0, `null`)

Table 1: Map of Dijkstra's algorithm results for last edge traversed to find each city for the first time.

(distance,last-edge)	Seq

Table 2: Table showing the progress of the priority queue in Dijkstra's algorithm.