**CS 311 HW7 Graph Algorithms Part 1**

**MST and Shortest Path (based on week 11 – 12)**

**==========================================================**

**DUE: Week 14 Tuesday at the beginning of class**

**TOTAL 32 points Your score:**

**\*NAME: Justin Goulet**

**\*DATE SUBMITTED: December 1, 2015**

**----------------------------------------------------**

**Purpose: To be able to follow graph algorithms**

**----------------------------------------------------**

**Review Questions from Week 12 and 13:**

1. **Using the following graph, do the Prim/Dijkstra algorithm**

**for finding a MST [12pts] Your score:**

Start with v1 and complete the following.

Give the grand total cost at the bottom.

1 1 4

v1 --- v2 ---- v3 ----- v4

3 | | 1

| |

v5 ----- v6

3

Tree: v1

Fringe: v2

Select v1-v2 with cost 1

Tree: v1 v2

Fringe: v3

**Select: v2-v3 with cost of 1**

**Tree: v1 v2 v3**

**Fringe: v4 v5**

**Select: v3-v5 with cost of 3**

**Tree: v1 v2 v3 v5**

**Fringe: v4 v6**

**Select: v3-v4 with cost of 4**

**Tree: v1 v2 v3 v5 v4**

**Fringe: v6**

**Select: v4-v6 with cost of 1**

**Highlight the MT edges in the above drawing.**

**TOTAL COST OF MT IS: 10**

**2. SHORTEST PATH [20pts] Your score:**

Use the following directed graph.

A to B is 4

A to F is 2

B to A is 1

B to C is 3

B to D is 4

C to A is 6

C to B is 3

C to D is 7

D to A is 6

D to E is 2

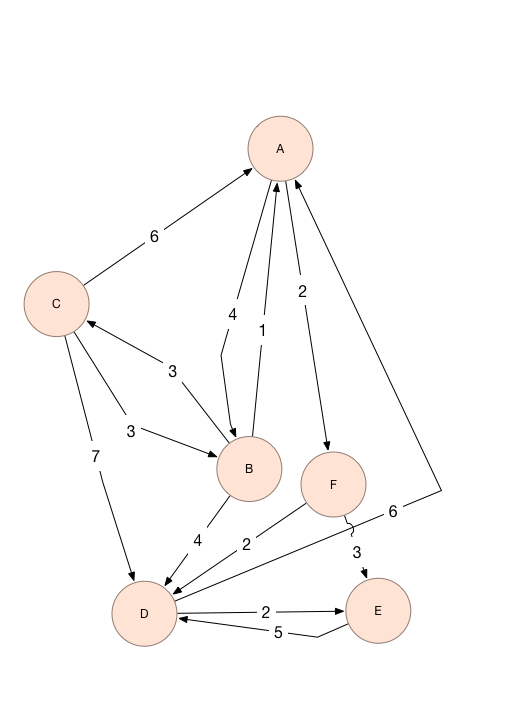
E to D is 5

F to D is 2

F to E is 3

**a) Do Dijkstra's shortest path algorithm starting with C**

**ending with E.**

****

Trace the algorithm using **the same format** as I used in Notes-12A. (Remember the tables???)

**Note that you may stop as soon as E is added to the Tree.**

|  |  |  |  |
| --- | --- | --- | --- |
| Node | Type | Distance To | Previous Node |
| A | T | 0 | Null |
| B | T | 4 | A |
| C | T | 7 | B |
| D | T | 4 | F |
| E | T | 5 | F |
| F | T | 2 | A |

New Tree: { A, F, B, D, E, C }

How was this discovered?

* We start by drawing our tree (for the purpose of this exercise to better view our example).
* Next, choose a starting node. I chose A because it has at least one outbound edge.
* Add ‘A’ to our tree. [A]
* Discover each of the outbound edges. [B(4), F(2)].
* Fill in the corresponding distances to each node with the previous node of ‘A’.
* Next, determine the shortest edge of the Fringe (F or B). In our case, F is closer with a distance of 2.
* Add this element to our tree. [A, F]
* Discover each of the outbound edges [D(2+2), E(3+2)]
* Fill in the corresponding distances to each node with the previous node of ‘F’. If a distance already exists, only replace the value if the new distance is less than the currently existing one.
* Next, determine our next working fringe from our temporary list [B, D, E] Note that the next working fringe should have the least distance
* Since our case is B, add to ‘B’ to our tree. [A, F, B]
* Discover each outbound edge [C(3+4), A(1)]
* Fill in the corresponding distances. Notice how we already found a shorter path to A (0) because we started on it. We can;t get a shorter path than 0, so the distance is not changed from the original.
* Next, determine the next working Fringe. It will be D.
* Add ‘D’ to our tree [A, F, B, D]
* Discover each outbound edge [A(6), E(9)].
* Notice how A is already in our tree. We ignore that one
* Fill in the corresponding data for E.
* Next, determine the next working Fringe. This time, it will be E.
* Add ‘E’ to our tree [A, F, B, D, E]
* Discover each outbound edge and update the table
* There aren't any outbound edges that can be updated because the only outbound edge comes from a previous node that is listed as a greater distance
* Our remaining Fringe is C. Add C to the tree

Starting with ‘C’

|  |  |  |  |
| --- | --- | --- | --- |
| Node | Type | Distance To | Previous Node |
| A | T | 4 | B |
| B | F | 3 | C |
| C | T | 0 | Null |
| D | T | 7 | B |
| E | T | 9 | F |
| F | T | 6 | A |

[C, B, A, F, D, E]

**b) What was the path from C to E found by the algorithm????**

**How did you determine that (what information from the table did**

**you use?)**

Shortest path from C to E is

C -> B -> A -> F -> E

The information was gathered using the set of data from the previous nodes. I worked backwards from E to C.

**Submit these files:**

* **This assignment sheet with your answers**