

COMPSCI/SFWRENG 2C03  
Data Structures and Algorithms

Ryszard Janicki  
McMaster University

## Assignment 3

Name: Hishmat Salehi

MacId: Salehh6

Student number: 400172262

1. a. 0:  $5 \rightarrow 2 \rightarrow 6$   
1:  $4 \rightarrow 8 \rightarrow 11$   
2:  $5 \rightarrow 6 \rightarrow 0 \rightarrow 3$   
3:  $10 \rightarrow 6 \rightarrow 2$   
4:  $1 \rightarrow 8$   
5:  $0 \rightarrow 10 \rightarrow 2$   
6:  $2 \rightarrow 3 \rightarrow 0$   
7:  $8 \rightarrow 11$   
8:  $1 \rightarrow 11 \rightarrow 7 \rightarrow 4$   
9:  
10:  $5 \rightarrow 3$   
11:  $8 \rightarrow 7 \rightarrow 1$

- b. Adjacency Matrix:

```
0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1,
1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0,
0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0,
```

2.

a)

dfs(0)

dfs(5)

check 0

↑

dfs(10)

check 5

↓

dfs(3)

check 10

↓

dfs(4)

dfs(2)

check 5

check 6

check 0

check 3

2 done

check 3

check 0

6 done

check 2

3 done

10 done

check 2

5 done

check 2

check 6

0 done

b)

0

↓

5

↓

10

↓

13

↓

6

↓

2

3.

a)

bfs(0)

bfs(5)

check 0

bfs(10)

check 5

check 3

check 2

bfs(2)

check 5

check 6

check 0

bfs(3)

check 10

check 6

check 2

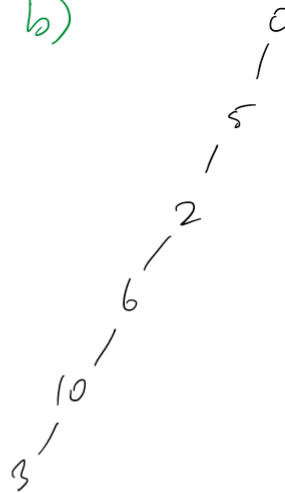
bfs(6)

check 2

check 3

check 0

b)



4. Consider by contradiction that the edge of maximum weight in the cycle  $C$ , edge  $e$ , belongs to the MST of the graph. Since MSTs do not contain cycles there is at least one edge in  $C$  that is not in the MST. Let's call one of these edges  $f$ . Now add  $f$  to the MST. There is now a cycle in the MST. Since  $e$  has the maximum weight in the cycle  $C$  and all edge weights are distinct, it means that  $\text{weight}(f) < \text{weight}(e)$ . Removing the edge  $e$  after having added the edge  $f$  would generate a new MST' with total weight less than the total weight in MST, contradicting its minimality.

5. a. 0:  $6 \rightarrow 5$   
 1:  
 2:  $0 \rightarrow 3$   
 3:  $10 \rightarrow 6$   
 4: 1  
 5:  $10 \rightarrow 2$   
 6: 2  
 7:  $8 \rightarrow 11$   
 8:  $1 \rightarrow 4$   
 9:  
 10: 3  
 11: 8

b. Adjacency Matrix:

```
0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1,
0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
```

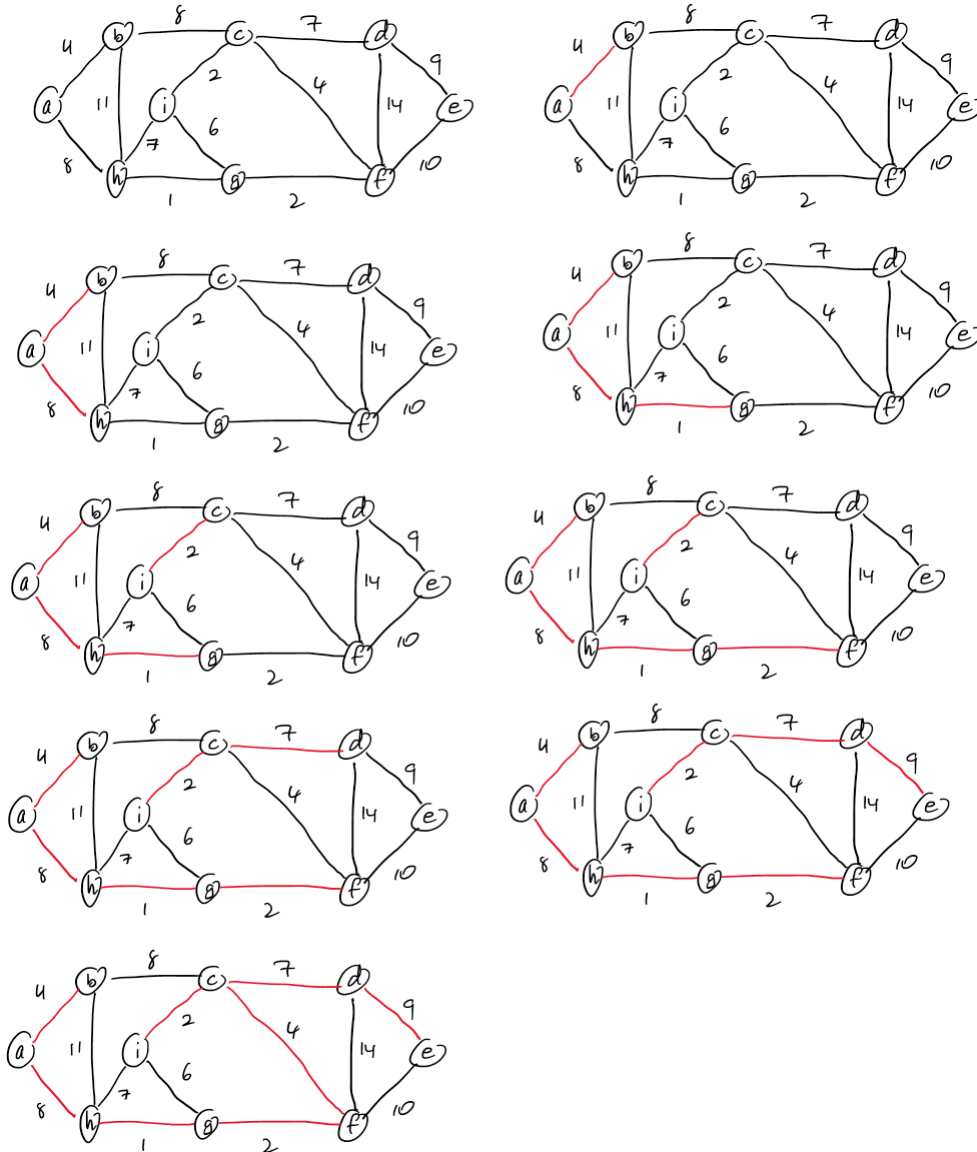
6. [TODO: Show steps pg. 589] It's strongest component is 0 2 3 5 6 10.

7. Topological order:

p - n - o - s - m - r - u - y - v - w - z - q - t - x

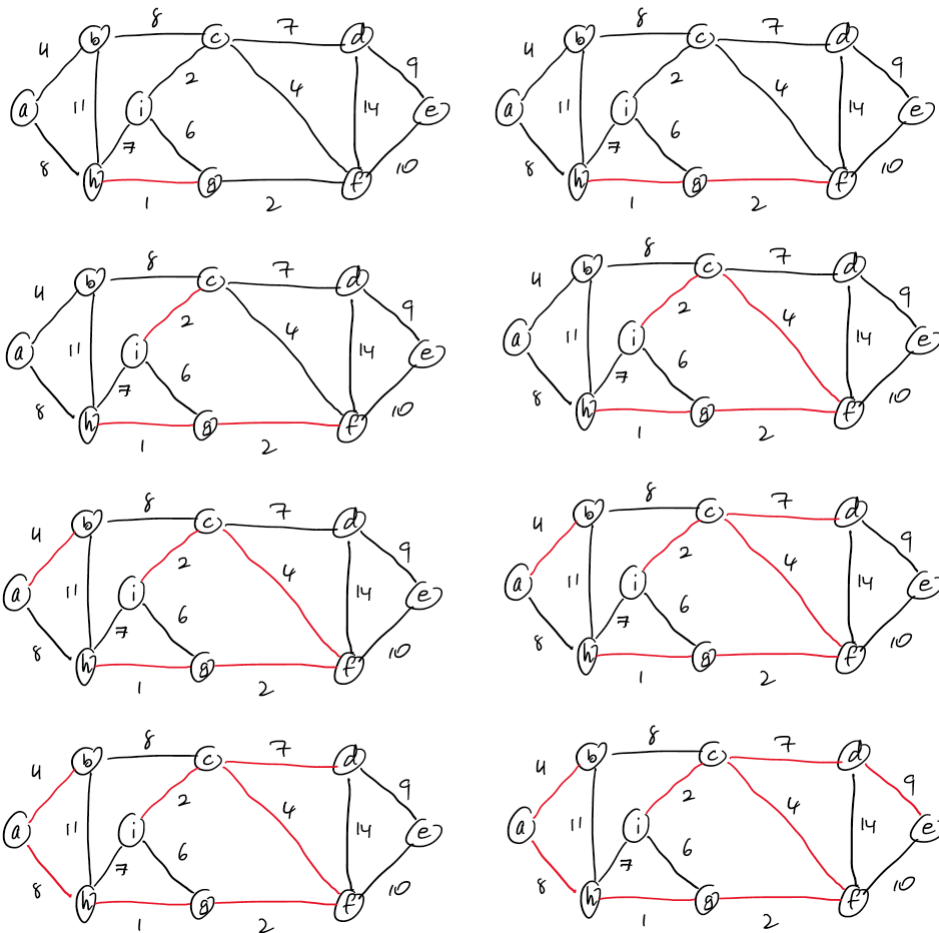
8. Suppose there are two minimum trees, A and B. Let  $e$  be the edge in just one of A,B with the smallest cost. Suppose it is in A but not B. Suppose  $e$  is the edge PQ. Then B must contain a path from P to Q which is not simply the edge  $e$ . So if we add  $e$  to B, then we get a cycle. If all the other edges in the cycle were in A, then A would contain a cycle, which it cannot. So the cycle must contain an edge  $f$  not in A. Hence, by the definition of  $e$  (and the fact that all edge-costs are different) the cost of  $f$  must be greater than the cost of  $e$ . So if we replace  $f$  by  $e$  we get a spanning tree with smaller total cost. Contradiction.

9. a. Minimum spanning tree with Greedy Algorithm:

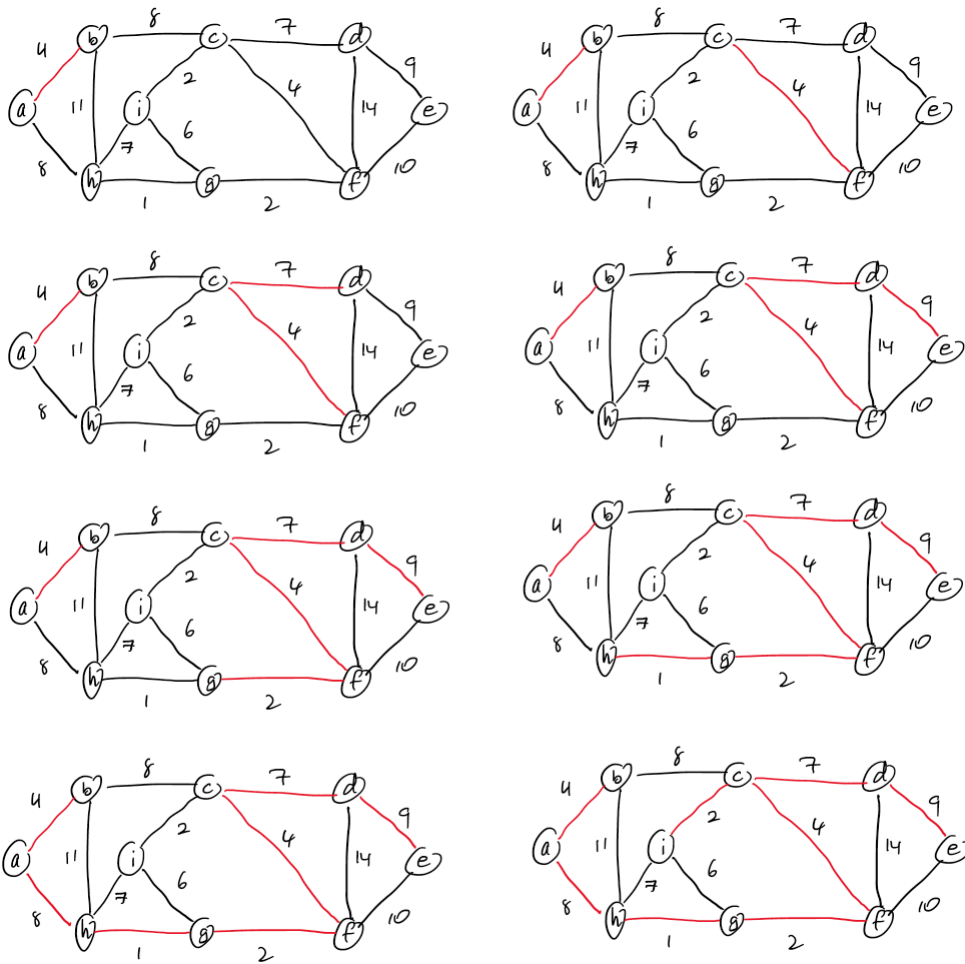


b. Minimum spanning tree with Kruskal's Algorithm:

Monday, April 6, 2020  
3:16 AM



c. Minimum spanning tree with Prim's Algorithm:



10. a to a (0.00)  
a to b (2.00) a→b 2.00  
a to c (6.00) a→j 4.00 j→c 2.00  
a to d (6.00) a→b 2.00 b→f 3.00 f→d 1.00  
a to e (8.00) a→l 5.00 l→e 3.00  
a to f (5.00) a→b 2.00 b→f 3.00  
a to h (5.00) a→h 5.00  
a to i (5.00) a→j 4.00 j→i 1.00  
a to j (4.00) a→j 4.00  
a to k (7.00) a→b 2.00 b→f 3.00 f→d 1.00 d→k 1.00  
a to l (5.00) a→l 5.00
11. s to s ( 0.00)  
s to t ( 2.00) s→y 7.00 y→x -3.00 x→t -2.00  
s to x ( 4.00) s→y 7.00 y→x -3.00  
s to y ( 7.00) s→y 7.00  
s to z (-2.00) s→y 7.00 y→x -3.00 x→t -2.00 t→z -4.00

```

12. public double diameter(EdgeWeightedDigraph edgeWeightedDigraph) {
    double diameter = Double.NEGATIVE_INFINITY;

    for (int v = 0; v < edgeWeightedDigraph.V(); v++) {
        DijkstraSP dijkstraSP = new DijkstraSP(edgeWeightedDigraph, v);

        for (int v2 = 0; v2 < edgeWeightedDigraph.V(); v2++) {
            if (dijkstraSP.distTo(v2) > diameter) {
                diameter = dijkstraSP.distTo(v2);
            }
        }
    }

    return diameter;
}

```

13. r to r (0.00) r to s (5.00) r-<sub>i</sub>s 5.00 r to t (3.00) r-<sub>i</sub>t 3.00 r to x (10.00) r-<sub>i</sub>t 3.00 t-<sub>i</sub>x 7.00 r to y (7.00) r-<sub>i</sub>t 3.00 t-<sub>i</sub>y 4.00 r to z (5.00) r-<sub>i</sub>t 3.00 t-<sub>i</sub>z 2.00

14. a. Give a trace for LSD string sort for the keys:  
no is th ti fo al go pe to co to th ai of th pa

input	d=1	d=0	output
no	pa	ai	ai
is	pe	al	al
th	of	co	co
ti	th	fo	fo
fo	th	go	go
al	th	is	is
go	ti	no	no
pe	ai	of	of
to	al	pa	pa
co	no	pe	pe
to	fo	th	th
th	go	th	th
ai	to	th	th
of	co	ti	ti
th	to	to	to
pa	is	to	to

b. Give a trace for MSD string sort for the keys:  
no is th ti fo al go pe to co to th ai of th pa

input	--	--			output
no	al	ai	ai	ai	ai
is	ai	al	al	al	al
th	co	--	co	co	co



ti	fo	co	fo	fo	fo
fo	go	fo	go	go	go
al	is	go	is	is	is
go	no	is	no	no	no
pe	of	no	of	of	of
to	pe	of	--	pa	pa
co	pa	pe	pa	pe	pe
to	th	pa	pe	--	th
th	ti	th	--	th	th
ai	to	ti	th	th	th
of	to	to	ti	th	ti
th	th	to	to	ti	to
pa	th	th	to	to	to
	--	th	th	to	
			th	--	

- c. Give a trace for MSD string sort for the keys:  
 now is the time for all good people to come to the aid of

input	---	---			output
now	all	aid	aid	aid	aid
is	aid	all	all	all	all
the	come	---	come	come	come
time	for	come	for	for	for
for	good	for	good	good	good
all	is	good	is	is	is
good	now	is	now	now	now
people	of	now	of	of	of
to	people	of	people	people	people
come	the	people	---	---	the
to	time	the	the	the	the
the	to	time	the	the	time
aid	to	to	time	---	to
of	the	to	to	time	to
	----	the	to	to	
			---	to	

15.

16. Failure Function:

P = abbababaaabab

j	0	1	2	3	4	5	6	7	8	9	10	11	12
P[j]	a	b	b	a	b	a	b	a	a	a	b	a	b
f(j)	0	0	0	1	2	1	2	1	1	1	2	1	2

Solution using Knuth-Morris-Pratt algorithm:

abaababababbababaaabababababb

123

abbababaaabab

45

abbababaaabab

678

abbababaaabab

-----

a b a a b a b a a b a b b a b a b a a a b a b a a b a a b a b b

9 1011

a b b a b a b a a a b a b

-----

a b a a b a b a a b a b b a b a b a a a b a b a a b a a b a b b

12131415161718192021222324

a b b a b a b a a a b a b

Final Answer:

abaababababbababaaabababababb

abbababaaabab

The algorithm performs 24 character comparisons, which are indicated with numerical labels.

17.

18.

19.

20.

21.