CS300 HOMEWORK 5 Tuğcan Barbin 25168

1-)dijkstra algorithm:

1-1)

known S(S,0)

not sure or unknown A(S,3), B(S,2), C(?,∞), D(?,∞), E(?,∞), F(?,∞), G(?,∞)

1-2)

known S(S,0), B(S,2)

not sure or unknown A(S,3), C(?,∞), D(B,3), E(?,∞), F(?,∞), G(?,∞)

1-3)

known S(S,0), B(S,2), A(S,3)

not sure or unknown C(?,∞), D(B,3), E(A,5), F(?,∞), G(?,∞)

1-4)

known S(S,0), B(S,2), A(S,3), D(B,3)

not sure or unknown C(?,∞), CHANGED E(D,4) , F(D,5) , G(D,5)

1-5)

known S(S,0), B(S,2), A(S,3), D(B,3), E(D,4)

not sure or unknown C(?,∞) , F(D,5) , G(D,5)

1-6)

known S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5)

not sure or unknown C(F,6) , G(D,5)

1-7)

known S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5), G(D,5)

not sure or unknown C(F,6)

1-8)

known S(S,0), B(S,2), A(S,3), D(B,3), E(D,4), F(D,5), G(D,5), C(F,6)

2-)Prim's minimum spanning tree algorithm:

2-1)

tree: S

route can be SB(2), SA(3)

2-2) tree S->B

route can be SA(3), BD(1), BA(2), BS(3)

***BS(3) IS REMOVED FROM PATH

2-3) tree S->B->D

route can be SA(3), BA(2), DE(1), DF(2), DG(2)

2-4) tree S->B->D->E

route can be SA(3), BA(2), DF(2), DG(2),EF(2), EG(3)

2-5) tree S->B->D->E

B->A

route can be SA(3), DF(2), DG(2),EF(2), EG(3)

***AE(2) AND AD(1) AND SA(3) ARE REMOVED FROM PATH

D->F

2-6)tree S->B->D->E

B->A

route can be DG(2),EF(2), EG(3), FC(1)

D->F->C

2-6)tree S->B->D->E

B->A

route can be DG(2),EF(2), EG(3)

***CB(4) IS REMOVED

D->G

D->F->C

2-6)tree S->B->D->E

B->A

route can be EF(2), EG(3)

*** EF(2) AND EG(3) AND GF(2) İS REMOVED FROM PATH

FINAL TREE IS

D---->F---->C

D---->G

S---->B---->D---->E

B---->A

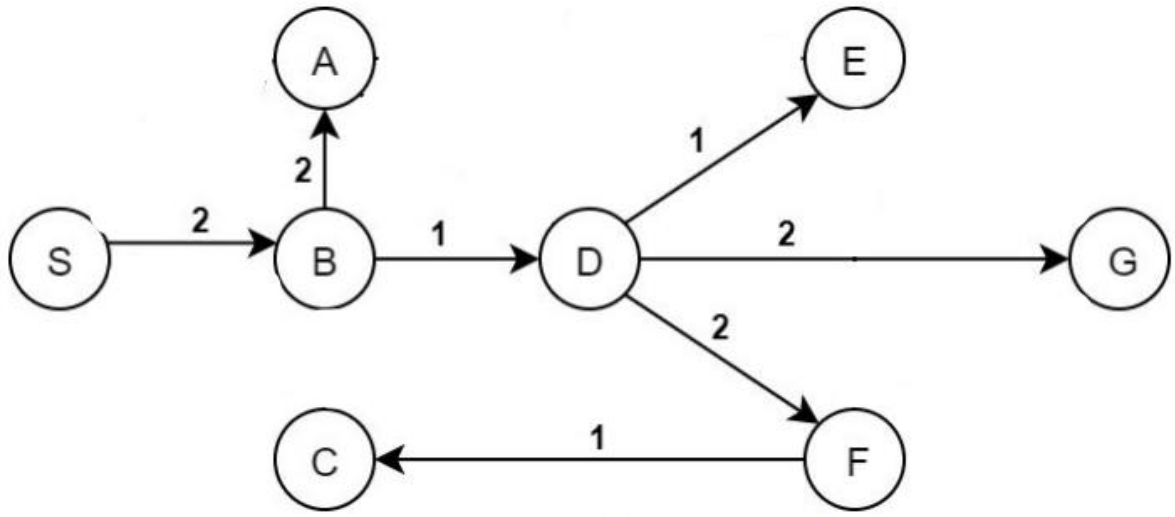


Figure 1: A directed weighted graph.

3-)KRUSKAL's minimum spanning tree algorithm

3-1)

trees:-

edges:-

possible edges: SA(3), SB(2), BS(3), BA(2),BD(1),AD(1), AE(2),EG(3), EF(2), GF(2), FC(1), CB(4), DE(1), DG(2), DF(2)

thrown edges:-

3-2)

trees: TREE1: D E

edges: DE(1)

possible edges: SA(3), SB(2), BS(3), BA(2),BD(1),AD(1), AE(2),EG(3), EF(2), GF(2), FC(1), CB(4), DG(2), DF(2)

thrown edges:

3-3)

trees: TREE1: B D E

edges: DE(1), BD(1)

possible edges: SA(3), SB(2), BS(3), BA(2),AD(1), AE(2),EG(3), EF(2), GF(2), FC(1), CB(4), DG(2), DF(2)

thrown edges:

3-4)

trees: TREE1: B D E  TREE3: F C

edges: DE(1), BD(1), FC(1)

possible edges: SA(3), SB(2), BS(3), BA(2),AD(1), AE(2),EG(3), EF(2), GF(2), CB(4), DG(2), DF(2)

thrown edges:

3-5)

trees: TREE1: B D E  TREE3: F C

edges: DE(1), BD(1), FC(1)

possible edges: SA(3), SB(2), BS(3), BA(2), AE(2),EG(3), EF(2), GF(2), CB(4), DG(2), DF(2)

thrown edges: AD(1)

3-6)

trees: TREE1: S B D E  TREE3: F C

edges: DE(1), BD(1), FC(1), SB(2)

possible edges: SA(3), BS(3), BA(2), AE(2),EG(3), EF(2), GF(2), CB(4), DG(2), DF(2)

thrown edges: AD(1)

3-7)

trees: TREE1: S B A D E  TREE3: F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2)

possible edges: SA(3), BS(3), AE(2),EG(3), EF(2), GF(2), CB(4), DG(2), DF(2)

thrown edges: AD(1)

3-8)

trees: TREE1: S B A D E  TREE3: F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2)

possible edges: SA(3), BS(3),EG(3), EF(2), GF(2), CB(4), DG(2), DF(2)

thrown edges: AD(1), AE(2)

3-9)

trees: TREE1: S B A D E G  TREE3: F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2)

possible edges: SA(3), BS(3),EG(3), EF(2), GF(2), CB(4), DF(2)

thrown edges: AD(1), AE(2)

3-10)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: SA(3), BS(3),EG(3), EF(2), GF(2), CB(4)

thrown edges: AD(1), AE(2)

*****DF(2) CONNECTED THE TWO TREES

3-10)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: SA(3), BS(3),EG(3), GF(2), CB(4)

thrown edges: AD(1), AE(2),EF(2)

3-11)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: SA(3), BS(3),EG(3), CB(4)

thrown edges: AD(1), AE(2),EF(2),GF(2)

3-12)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: SA(3), BS(3), CB(4)

thrown edges: AD(1), AE(2),EF(2),GF(2),EG(3)

3-12)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: BS(3), CB(4)

thrown edges: AD(1), AE(2),EF(2),GF(2),EG(3), SA(3)

3-13)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: CB(4)

thrown edges: AD(1), AE(2),EF(2),GF(2),EG(3), SA(3), BS(3)

3-14)

trees: TREE1: S B A D E G F C

edges: DE(1), BD(1), FC(1), SB(2), BA(2), DG(2), DF(2)

possible edges: -

thrown edges: AD(1), AE(2),EF(2),GF(2),EG(3), SA(3), BS(3),CB(4)

there could be many different combinations due to equal weighted paths i wanted it to be same with prim's. So, result is same with the prim's.

4-) Breadth-first traversal

I tried to do this operation with weighted path but it is not worked out for me

problem has no explainings about it but i thought that it must be a unweighted path because in the lecture the breadth-first is used on unweighted path

4-1)S is 0 away from itself so it is root

4-2)visiting childs of S which are 1 away from S (B,A)

4-3)visiting childs of B which are 2 away from S (D) (A is already child of S, S is already root)

4-4)visiting childs of A which are 2 away from S (E) (D is already child of B)

4-5)visiting childs of D which are 3 away from S (F,G) (E is already child of A)

4-6)visiting childs of E which are 3 away from S - (F and G already child of D)

4-7)visiting childs of F which are 4 away from S (C)

4-8)visiting childs of C which are 5 away from S - (B is already child of S)

5-)

a) Depth-first traversal

a-1)S is our root go to left child of S (B)

a-2)go to left child of B (D)

a-3)go to left child of D (F)

a-3)go to left child of F (C)

a-4)then it returns to F because there is no left child of C that is not visited

a-5)then it returns to D because there is no left child of F that is not visited

a-6)go to left child of D (G)

a-7)then it returns to D because there is no left child of G that is not visited

a-8)go to left child of D (E)

a-9)then it returns to D because there is no left child of E that is not visited

a-10)then it returns to B because there is no left child of D that is not visited

a-11)go to left child of B (A)

a-12)then it returns to B because there is no left child of A that is not visited

a-13)then it returns to S because there is no left child of B that is not visited

a-14)then it stops because there is no left child of S that is not visited

b)post order numbers of vertexes

A =

c)pre order numbers of vertexes

S=1, B=2, D=3, F=4, C=5, G=6, E=7, A=8

d)post order numbers of vertexes

C=1, F=2, G=3, E=4, D=5, A=6, B=7, S=8

tree arcs are shown as red in graph which are;

SB, BD, BA, DE, DG, DF, FC

cross arcs are shown as blue in graph which are;

AD, AE, EG, EF, GF

forward arcs are shown as green in graph which are;

SA

backward arcs are shown as black in graph which are;

CB

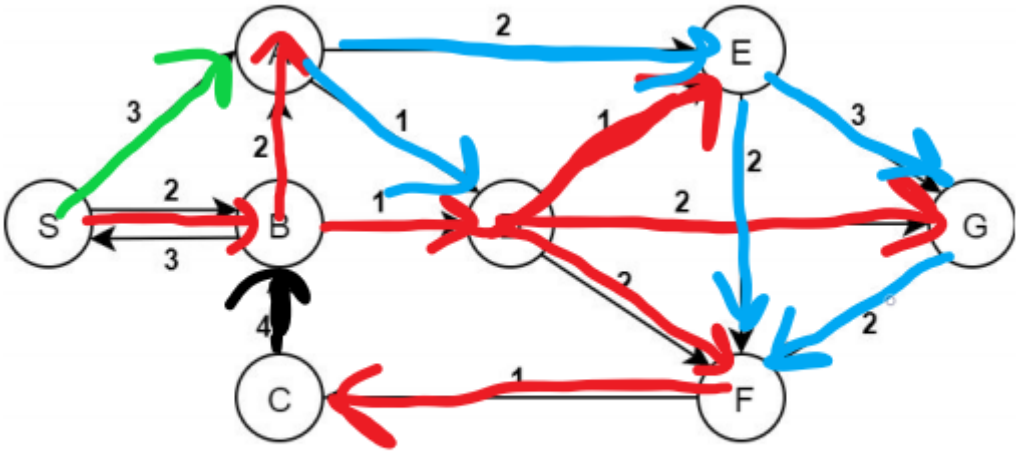


Figure 1: A directed weighted graph.

6-)TOPOLOGICAL SORT

STARTING FROM D , DFS USED

D E F G B A C

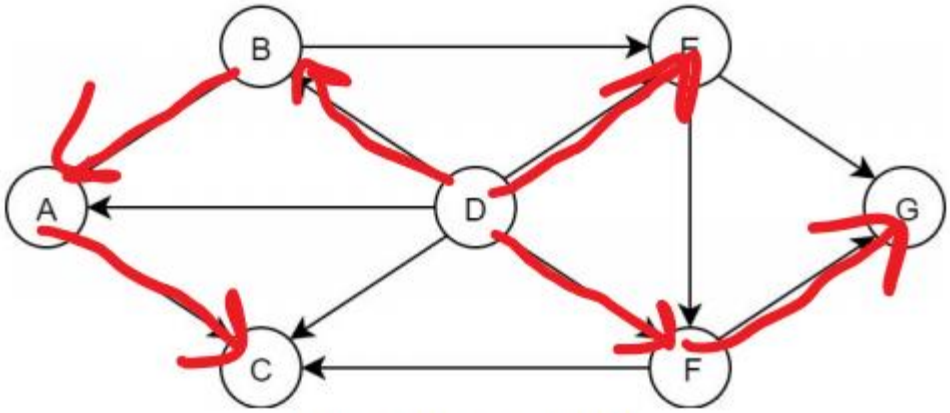


Figure 2: An example DAG.