COMP3711: Design and Analysis of Algorithms

Tutorial 10

HKUST

Question 1

Floyd-Warshall problem execution.
See external file.

Question 2

(CLRS)
Give an algorithm that takes as input a directed graph with positive edge weights, and returns the cost of the shortest cycle in the graph (if the graph is acyclic, it should say so).

Your algorithm should take time at most $O(n^3)$ time, where n is the number of vertices in the graph.

Solution

We can solve this by using the *Floyd-Warshall* algorithm for the all-pairs shortest path problem.

In $O(n^3)$ time run the Floyd-Warshall algorithm to find all of the values $d_{i,j}$, the costs of the min cost paths from v_i to v_j .
We claim that the answer to the problem is then

$$A = \min_{i,j}\left(d_{i,j} + d_{j,i}\right)$$

The running time is $O(n^3)$ for the Floyd-Warshall algorithm and $O(n^2)$ for the rest, so the full running time is $O(n^3)$.

It remains to prove correctness of the algorithm.
Let OPT be the real cost of a minimum cost cycle.

We need to show that $A = OPT$.

Solution

$A = \min_{i,j}\left(d_{i,j} + d_{j,i}\right)$ and OPT is the real cost of a minimum cost cycle.

A) Every term $d_{i,j} + d_{j,i}$ is the cost of *some* cycle -- start at v_i, follow path of length $d_{i,j}$ to v_j, then follow path of length $d_{j,i}$ back to v_i.
Since OPT is the minimum of all cycle costs,

$$OPT \le A.$$

B) Now suppose that we know some min-cost cycle C.
Let $v_{i'}, v_{j'}$ be any two points on the cycle.
Let $d'_{i',j'}$ and $d'_{j',i'}$ be the costs **on that cycle** for the paths from $v_{i'}$ to $v_{j'}$ and from $v_{j'}$ to $v_{i'}$. By definition

$$d'_{i',j'} \ge d_{i',j'} \text{ and } d'_{j',i'} \ge d_{j',i'}$$

=> $$OPT = cost(C) = d'_{i',j'} + d'_{j',i'} \ge d_{i',j'} + d_{j',i'} \ge A$$

Thus, $A = OPT$.

Question 3

In the class notes on the Floyd-Warshall Algorithm we said that it was possible to reduce the space requirement from $O(n^3)$ to $O(n^2)$ by not keeping all of the n. $n \times n$ matrices $D^{(i)}$, but instead, keeping only ONE matrix and reusing it.

We then wrote the code for doing that.

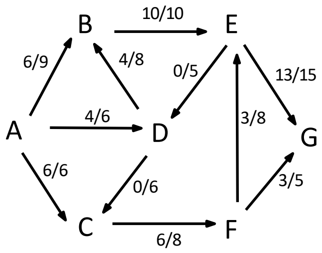
Why does this space-reduced code work and give the correct answer?

Solution

See additional powerpoint slides.

Question 4

Consider the given graph with flow values f and capacities c (f/c) as shown. $s = A$ and $t = G.$



Draw the residual graph.

Find an augmenting path.

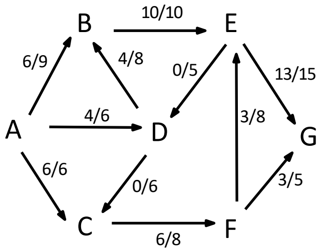
Show the new flow created by adding the augmenting path flow.

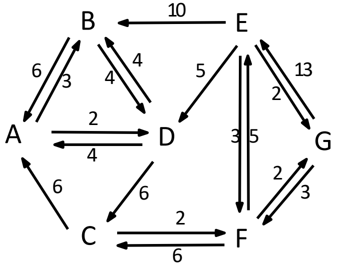
Is your new flow optimal?

Prove or disprove.

Solution

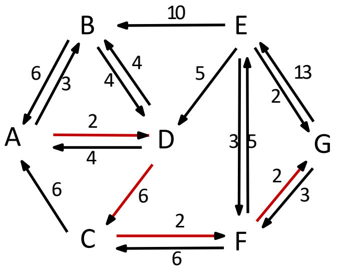
Draw, the residual graph.

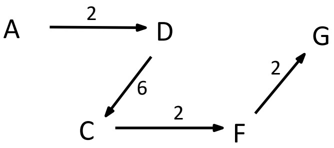




Solution

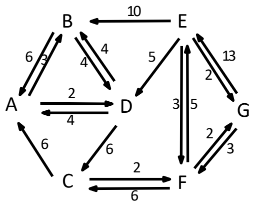
Find an augmenting path.

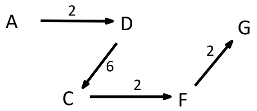


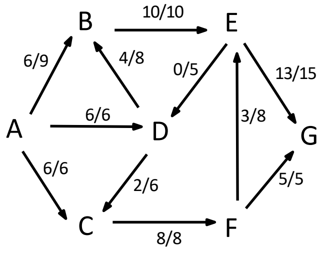


Solution

Show the new flow created by adding the augmenting path flow.

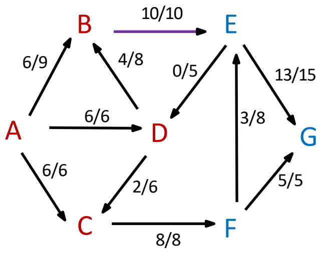


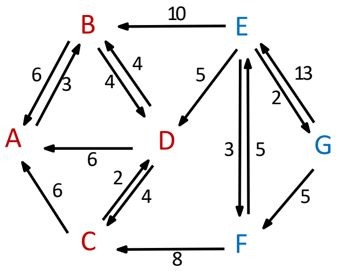




Solution

Is your new flow optimal? Prove or disprove.





It is optimal.

Consider the S, T cut defined by $S = \{A, B, C, D\}$, which is the set of nodes that can be reached from A in the residual graph of the new flow.
Then $T = \{E, F, G\}$.

Capacity of the cut is $C(S, T) = 10 + 8 = 18$ which is equal to $6 + 6 + 6$, exactly the value of the flow leaving A.

$\Rightarrow$ from the max-flow min-cut theorem, this is a maximal flow.

Question 5

Max-Flow in taught in class assumed a single source and single sink.

Suppose the flow network has

multiple sources $s_1, s_2, \dots, s_m$ and
multiple sinks. $t_1, t_2, \dots, t_n$ and

the goal is to move as much from all the sources to all of the sinks as possible.

a) Extend the flow properties and definitions to the multiple-source, multiple-sink problem.

b) Show that any flow in a multiple-source, multiple-sink flow network corresponds to a flow of identical value in the single-source, single-sink network obtained by adding a supersource and a supersink, and vice versa.

Solution

a) The definition becomes to find a flow f with maximum value

$$|f| = \sum_{i=1}^{m} \sum_{v} f(s_i, v) = \sum_{i=1}^{n} \sum_{v} f(v, t_i)$$

The flow conservation property becomes

$$\forall v \in V - \{s_1, s_2, \ldots, s_m, t_1, t_2, \ldots, t_n\},$$

$$\sum_{e\ out\ of\ v} f(e) = \sum_{e\ into\ v} f(e)$$

Solution

Obtain a flow network G' from G by adding

a supersource s with edges (s, s_i) for $1 \leq i \leq m$

and

a supersink t with edges (t_i, t), for $1 \leq i \leq n$

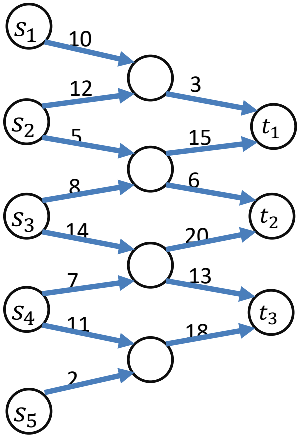
where the capacity of all these new edges are set to ∞.

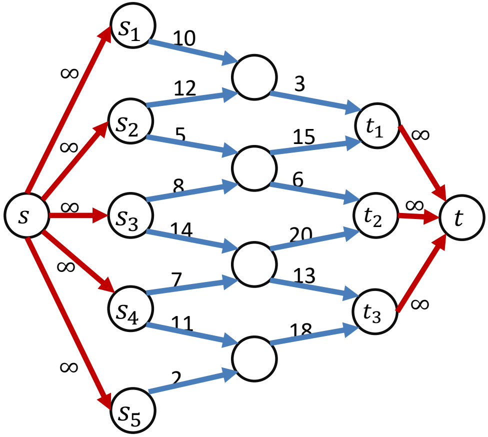
Then, any flow in G corresponds to a flow of identical value in G', and vice versa.

Solution

A multi-source multi-sink problem

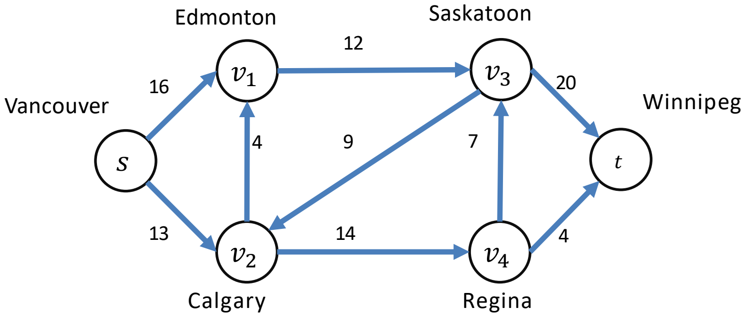
and its equivalent single-source single-sink version.





Question 6

Show the execution of the Edmonds-Karp algorithm on the following flow network.



Solution

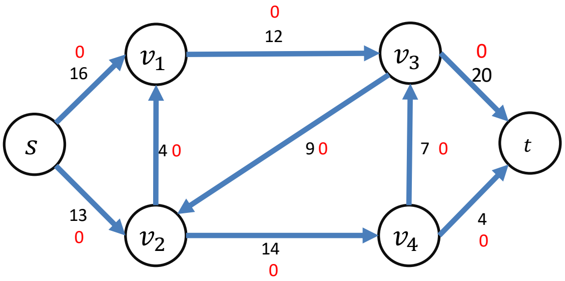
Recall that the Edmonds-Karp algorithm is the version of the Ford-Fulkerson method that always chooses a *shortest* (by number of edges) $s - t$ Augmenting Path in the residual graph G_f.

Let $d_f(s,t)$ denote the minimum number of edges in an $s - t$ Augmenting Path in the residual graph G_f.

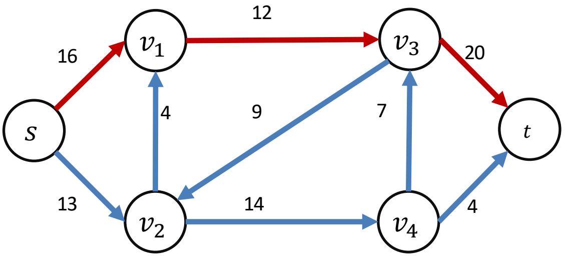
Recall that such a $s - t$ path with a minimum # of edges, can be found by running a BFS starting from *s*.

Solution

G:



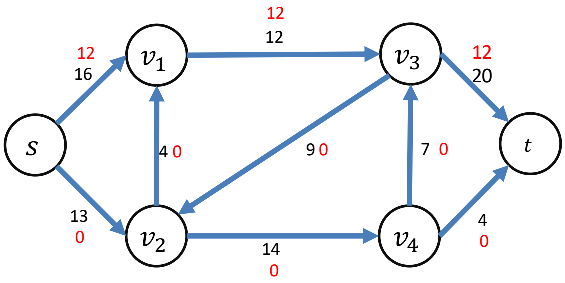
G_f:



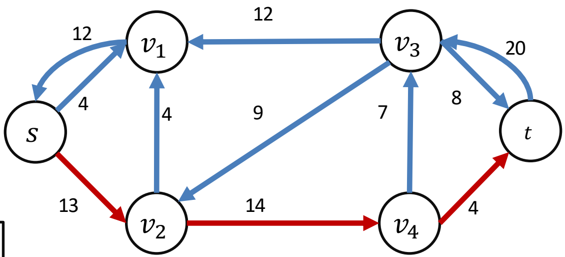
$\boxed{d_f(s,t) = 3}$

Solution

G:



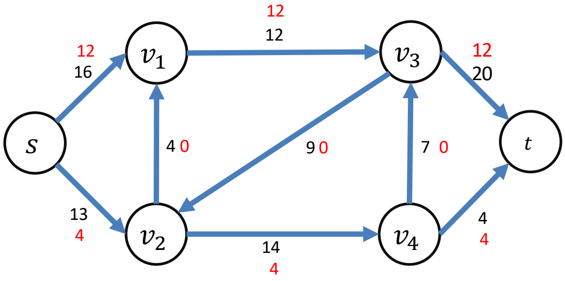
G_f:



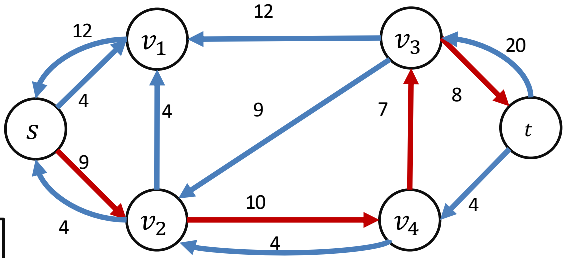
$$\boxed{d_f(s,t) = 3}$$

Solution

G:



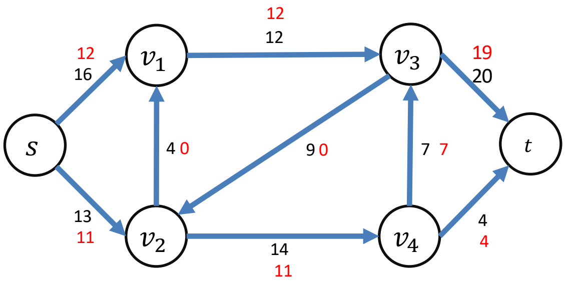
G_f:



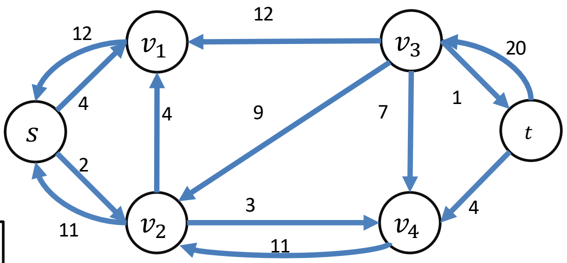
$$\boxed{d_f(s,t) = 4}$$

Solution

G:



G_f:

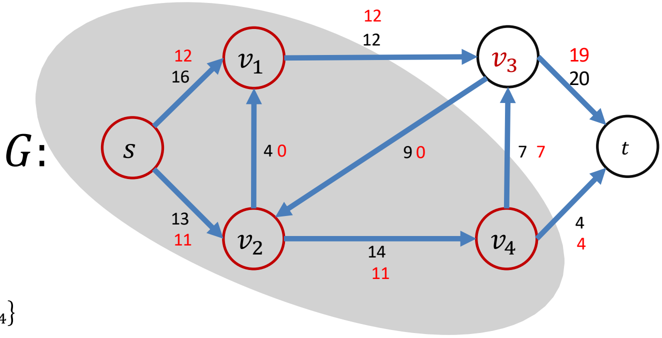


$$\boxed{d_f(s,t) = \infty}$$

Max flow value = 12+11= 23

Max cut capacity = 12+7+4 =23

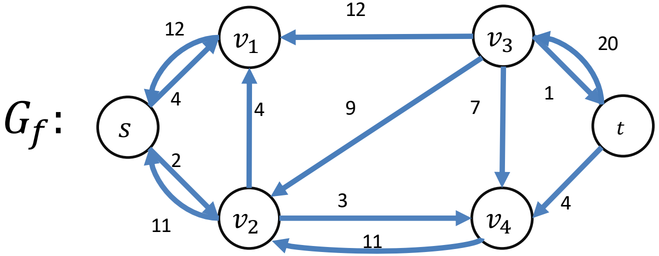
Solution



Max-Cut is

$S = \{s, v_1, v_2, v_4\}$

V-S= $\{t, v_3\}$



G_f:
s
v_1
v_2
v_3
v_4
t
12
4
12
4
9
7
1
20
2
11
3
11
4