طراحی الگوریتم

۱۲آذر ۹۸

ملکی مجد

Topic	Reference
Recursion and Backtracking	Ch.1 and Ch.2 JeffE
Dynamic Programming	Ch.3 JeffE and Ch.15 CLRS
Greedy Algorithms	Ch.4 JeffE and Ch.16 CLRS
Amortized Analysis	Ch.17 CLRS
Elementary Graph algorithms	Ch.6 JeffE and Ch.22 CLRS
Minimum Spanning Trees	Ch.7 JeffE and Ch.23 CLRS
Single-Source Shortest Paths	Ch.8 JeffE and Ch.24 CLRS
All-Pairs Shortest Paths	Ch.9 JeffE and Ch.25 CLRS
Maximum Flow	Ch.10 JeffE and Ch.26 CLRS
String Matching	Ch.32 CLRS
NP-Completeness	Ch.12 JeffE and Ch.34 CLRS

Maximum Flow

+a graph-theoretic definition of flow networks

*A **flow network*** $G = (V, E)$:

is a directed graph in which each edge $(u, v) \in E$ has a nonnegative ***capacity*** $c(u, v) \geq 0$.

If $(u, v) \notin E$,

we assume that $c(u, v) = 0$.

In Flow networks

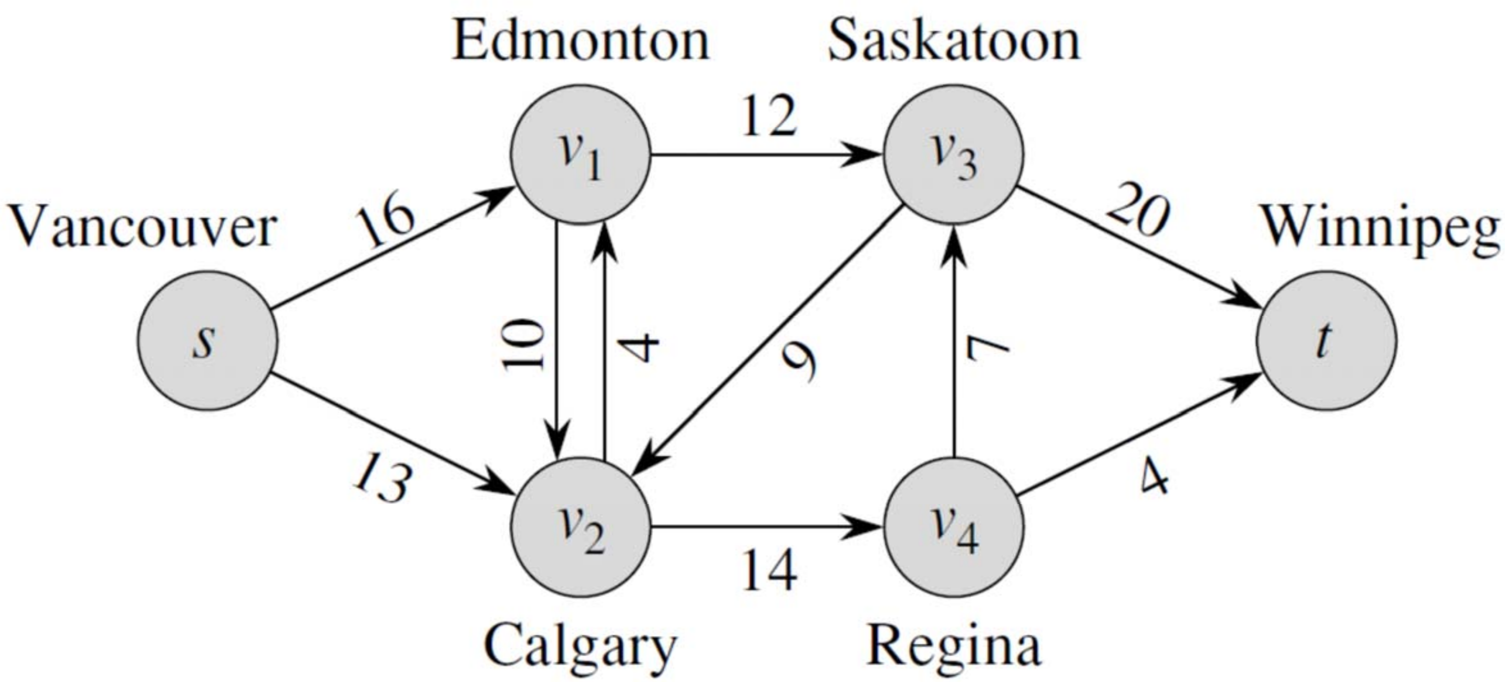
We distinguish : a ***source*** *s* and a ***sink*** *t*

we assume :

every vertex lies on some path from the source to the sink

The graph is therefore connected, and $|E| \geq |V| - 1$.

Example of netwrok



flow

A flow in G is a **real-valued function f** $: V \times V \rightarrow R$ that satisfies the following three properties:

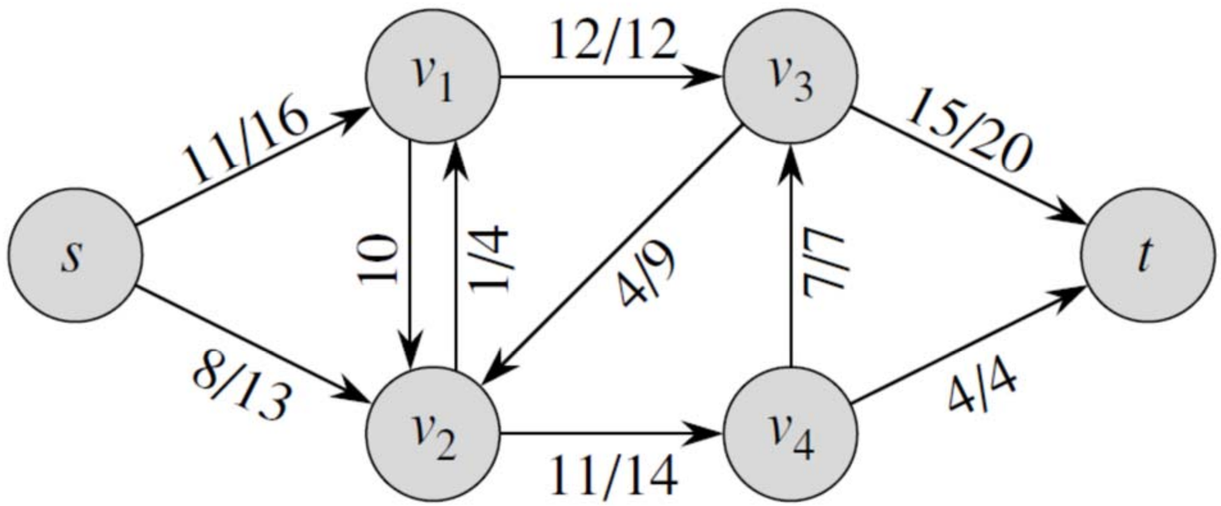
- **Capacity constraint:** For all $u, v \in V$, we require $f(u, v) \leq c(u, v)$.
- **Skew symmetry:** For all $u, v \in V$, we require $f(u, v) = -f(v, u)$.
- **Flow conservation:** For all $u \in V - \{s, t\}$, we require $\sum_{v \in V} f(u, v) = 0$

flow in equals **flow out** for vertex other than source and sink

The ***value*** of a flow ***f :***

total flow out of the source ($|f| = \sum_{v \in V} f(s, v)$)

A sample flow



maximum-flow problem

In the ***maximum-flow problem***, we are given a flow network G with source s and sink t, and we wish to find a flow of maximum value.

Networks with multiple sources and sinks

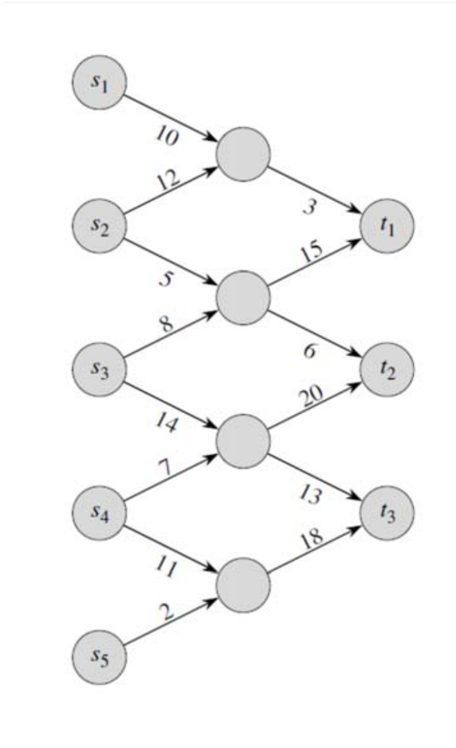
- This problem is no harder than ordinary maximum flow
- We can **reduce the problem** of determining a maximum flow in a network with multiple sources and multiple sinks to an ordinary maximum-flow problem

- **add**

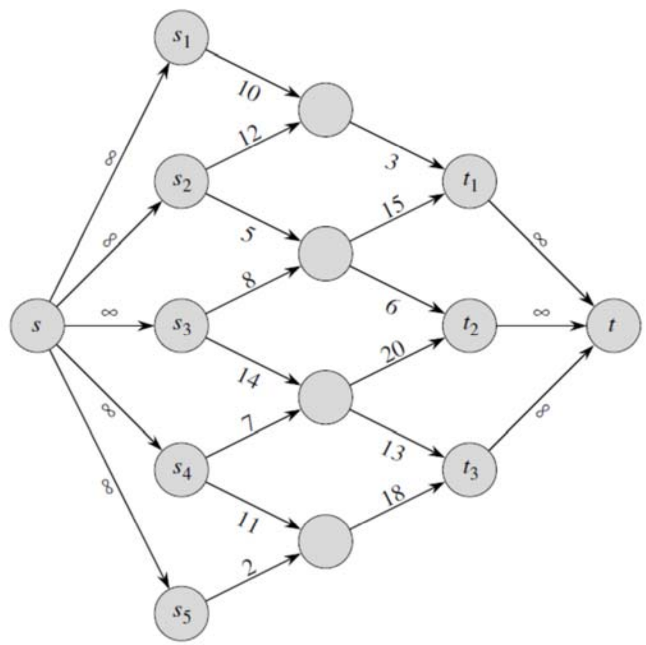
a ***supersource*** *s* and add a directed edge (s, s_i) with capacity $c(s, s_i) = \infty$ for each sourse s_i $(i = 1, 2, \ldots, m)$.

a ***supersink*** *t* and add a directed edge (t_i, t) with capacity $c(t_i, t) = \infty$ for each sink t_i $(i = 1, 2, \ldots, m)$.

A network with multiple sources and sinks



Convert to a network with one source and one sink



lemma

$$f(X, Y) = \sum_{x \in X} \sum_{y \in Y} f(x, y)$$

Let $G = (V, E)$ be a flow network, and let f be a flow in G. Then the following equalities hold:

1. For all $X \subseteq V$, we have $f(X, X) = 0$.
2. For all $X, Y \subseteq V$, we have $f(X, Y) = -f(Y, X)$.
3. For all $X, Y, Z \subseteq V$ with $X \cap Y = \emptyset$, we have the sums $f(X \cup Y, Z) = f(X, Z) + f(Y, Z)$ and $f(Z, X \cup Y) = f(Z, X) + f(Z, Y)$.

The value of flow

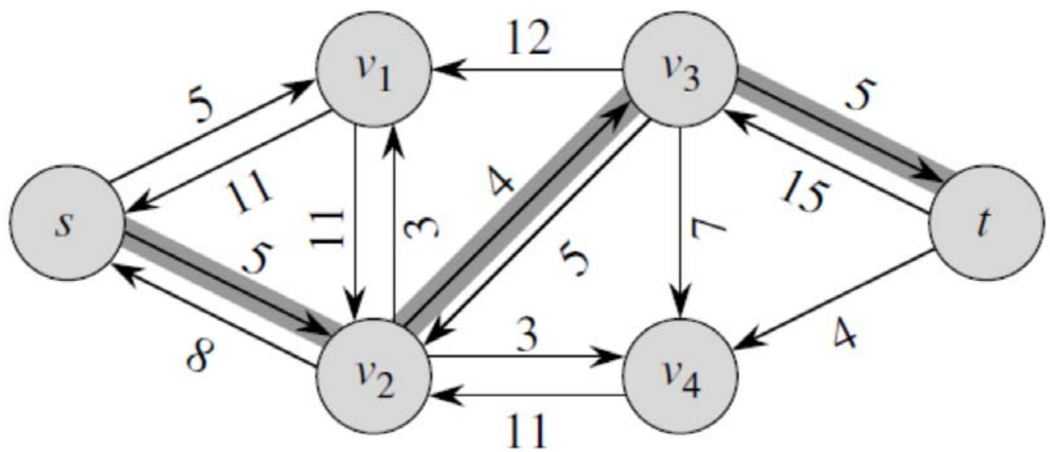
$$
\begin{aligned}
|f| &= f(s, V) \\
&= f(V, V) - f(V - s, V) \\
&= -f(V - s, V) \\
&= f(V, V - s) \\
&= f(V, t) + f(V, V - s - t) \\
&= f(V, t)
\end{aligned}
$$

The Ford-Fulkerson method

+for solving the maximum-flow problem

Augmenting paths

- a path from the source *s* to the sink *t* along which we can send more flow, and then augmenting the flow along this path



General method
How increase the value of flow

- FORD-FULKERSON-METHOD*(G, s, t)*

1 initialize flow *f* to 0

2 **while** there exists an augmenting path *p*

3 **do** augment flow *f* along *p*

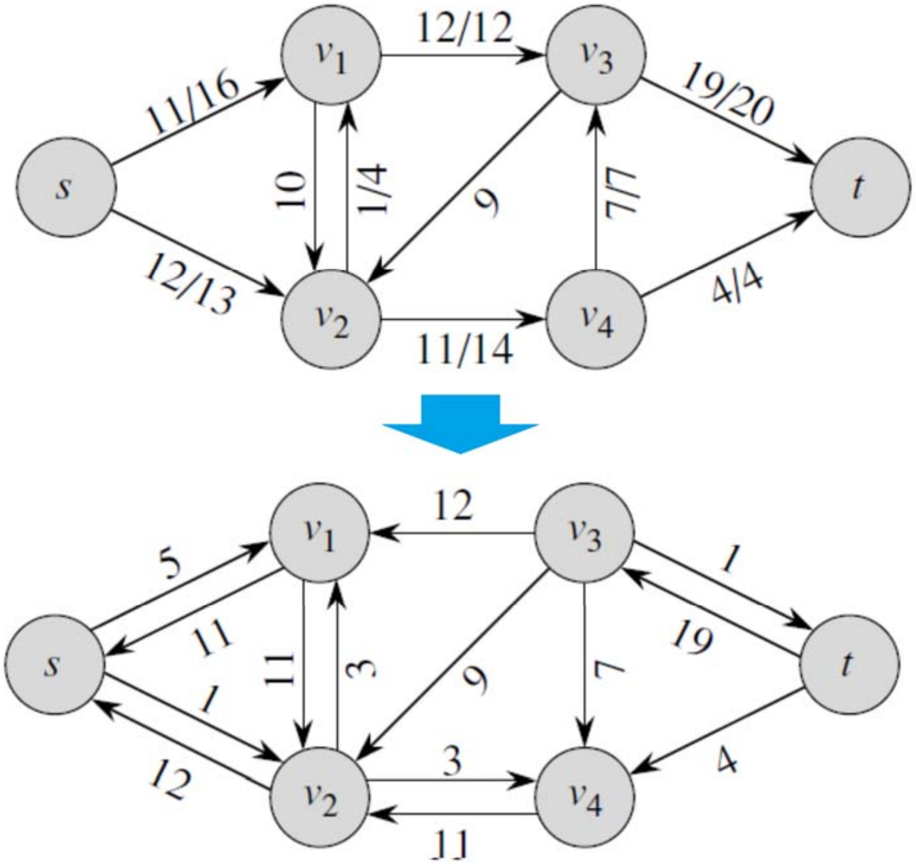
4 **return** *f*

Residual networks

- The amount of *additional* flow we can push from *u* to *v* before exceeding the capacity *c(u, v)* is the ***residual capacity*** of *(u, v)*, given by

$$c_f(u, v) = c(u, v) - f(u, v).$$

given a flow network and a flow, the residual network consists of edges that can admit more flow



v1
v2
v3
v4
s
t
12/12
11/16
19/20
10
1/4
9
7/7
12/13
4/4
11/14
12
5
11
1
19
11
3
9
7
1
3
12
4
11

- a flow is maximum if and only if its residual network contains no augmenting path.

how a flow in a residual network relates to a flow in the original flow network

Let $G = (V, E)$ be a flow network with source s and sink t, and let f be a flow in G. Let G_f be the residual network of G induced by f, and let f' be a flow in G_f. Then the flow sum $f + f'$ defined by equation (26.4) is a flow in G with value $|f + f'| = |f| + |f'|$.

residual capacity of an augmenting path

- the maximum amount by which we can increase the flow on each edge in an augmenting path *p*

$$c_f(p) = \min\{c_f(u, v) : (u, v) \text{ is on } p\}$$

residual capacity *of an* ***augmenting path*** *can add to the value of* ***flow***

Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Define a function $f_p : V \times V \to \mathbf{R}$ by

$$f_p(u, v) = \begin{cases} c_f(p) & \text{if } (u, v) \text{ is on } p \,, \\ -c_f(p) & \text{if } (v, u) \text{ is on } p \,, \\ 0 & \text{otherwise} \,. \end{cases} \tag{26.6}$$

Then, f_p is a flow in G_f with value $|f_p| = c_f(p) > 0$.

Corollary

Let $G = (V, E)$ be a flow network, let f be a flow in G, and let p be an augmenting path in G_f. Let f_p be defined as in equation (26.6). Define a function $f' : V \times V \to \mathbf{R}$ by $f' = f + f_p$. Then f' is a flow in G with value $|f'| = |f| + |f_p| > |f|$.

Definition of cut

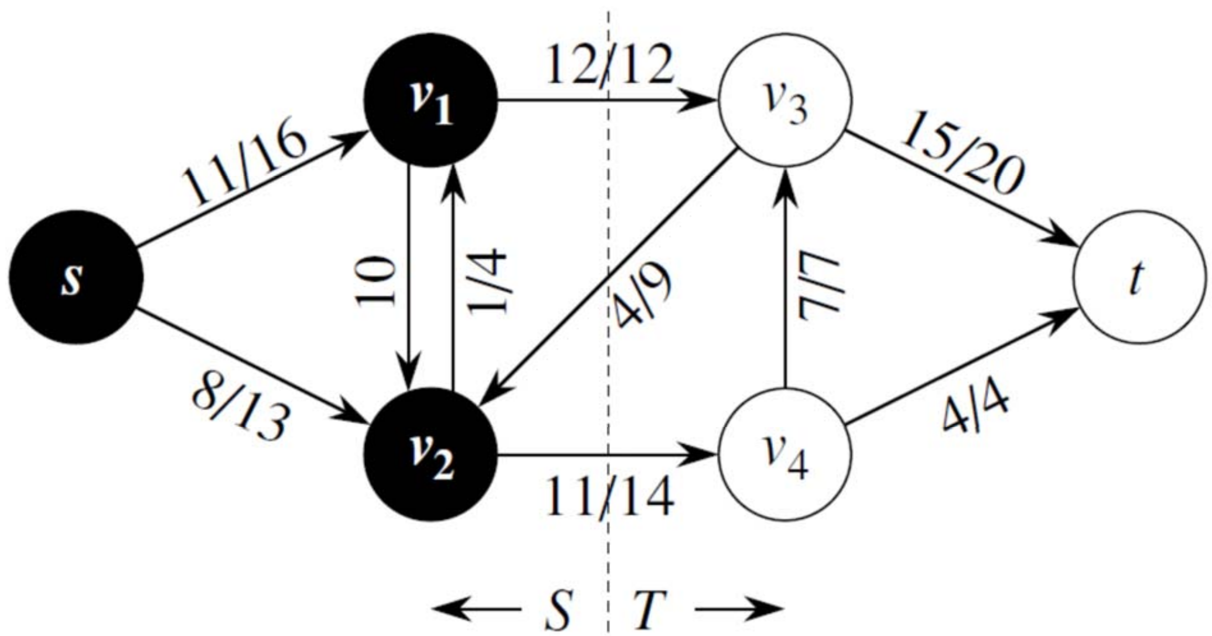
- A ***cut*** *(S, T)* of flow network $G = (V, E)$ is a partition of *V* into *S* and $T = V - S$ such that $s \in S$ and $t \in T$.

- If *f* is a flow, then the ***net flow*** across the cut *(S, T)* is defined to be

$$f(S, T).$$

- The ***capacity*** of the cut *(S, T)* is *c(S, T)*.
- A ***minimum cut*** of a network is a cut whose capacity is minimum over all cuts of the network.

A cut with
the net flow $f(S, T) = 19$, and
the capacity $c(S, T) = 26$.



- The value of any flow f in a flow network G is bounded from above by the capacity of any cut of G.

(can proved by definition of flow and cut)

Max-flow min-cut theorem

If f is a flow in a flow network $G = (V, E)$ with source s and sink t, then the following conditions are equivalent:

1. f is a maximum flow in G.
2. The residual network G_f contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.

basic Ford-Fulkerson algorithm

expands on the FORD-FULKERSONMETHOD

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ E[G]
     do f[u, v] ← 0
        f[v, u] ← 0
while there exists a path p from s to t in the residual network G_f
     do c_f(p) ← min {c_f(u, v) : (u, v) is in p}
        for each edge (u, v) in p
            do f[u, v] ← f[u, v] + c_f(p)
               f[v, u] ← −f[u, v]
```

Time complexity

- The running time of FORD-FULKERSON depends on how the augmenting path *p* in line 4 is determined.

- where is the maximum flow found by the algorithm:

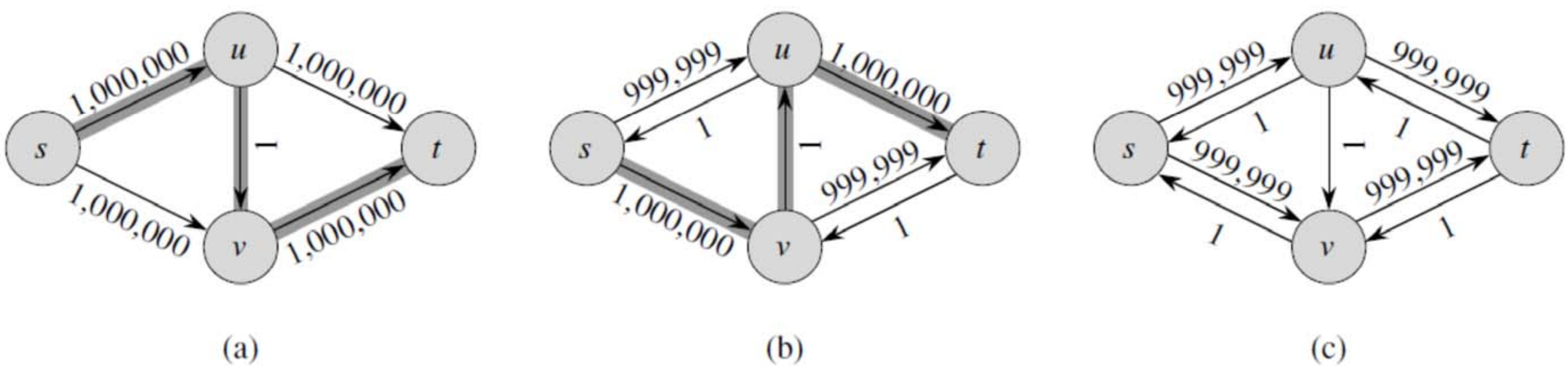
  a straightforward implementation runs in time $O(E\ |f^*|)$

- Prove Hint:

  the flow value increases by at least one unit in each iteration

$O(E\ |f^*|)$ can be bad!

- continue, choosing the augmenting path $s \rightarrow u \rightarrow v \rightarrow t$ in the odd-numbered iterations and the augmenting path $s \rightarrow v \rightarrow u \rightarrow t$ in the even-numbered iterations.



The Edmonds-Karp algorithm

- The bound on FORD-FULKERSON can be improved if we implement the computation of the augmenting path *p* in line 4 with a breadth-first search (each edge has unit distance (weight))

the augmenting path is a *shortest* path from *s* to *t* in the residual network

the running time of the Edmonds-Karp algorithm is $O(V\,E^2)$

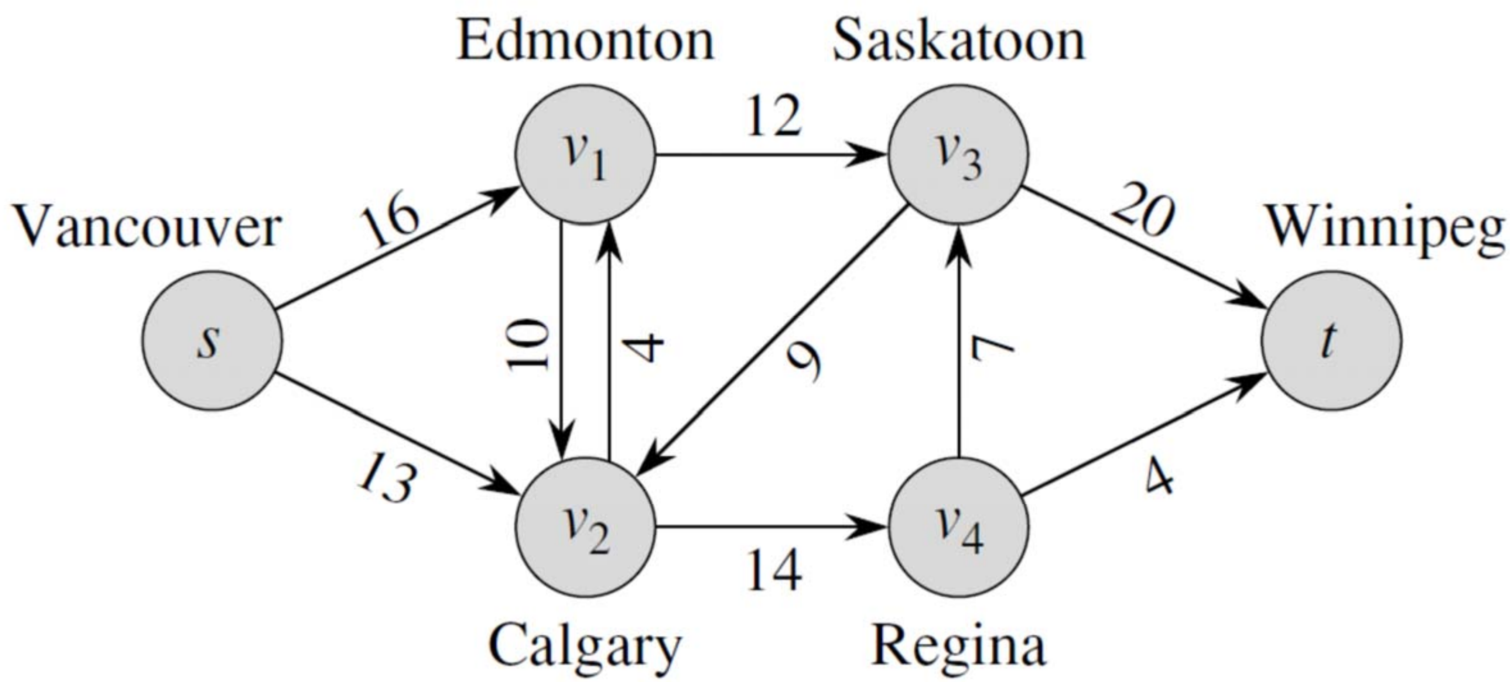
Lemma 26.8

If the Edmonds-Karp algorithm is run on a flow network $G = (V, E)$ with source s and sink t, then for all vertices $v \in V - \{s, t\}$, the shortest-path distance $\delta_f(s, v)$ in the residual network G_f increases monotonically with each flow augmentation.

Theorem 26.9

If the Edmonds-Karp algorithm is run on a flow network $G = (V, E)$ with source s and sink t, then the total number of flow augmentations performed by the algorithm is $O(VE)$.

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



Sample Problem

Suppose you are given a flow network G with ***integer*** edge capacities and an ***integer*** maximum flow f^* in G. Describe algorithms for the following operations:

(a) INCREMENT(e): Increase the capacity of edge e by 1 and update the maximum flow.

(b) DECREMENT(e): Decrease the capacity of edge e by 1 and update the maximum flow.

Both algorithms should modify f^* so that it is still a maximum flow, more quickly than recomputing a maximum flow from scratch.