Maximum Flow Problem

Input: Directed graph (flow network)

$G = (V, E)$

Capacities of edges $c : E \to \mathbb{Z}^+$

Special vertices s = source, t = sink $\in V$

Model: s produces a commodity

t consumes it

Commodity needs to flow on this network.

$f(u,v)$ = flow from u to v $\leq c(u,v)$ capacity of (u,v).

Problem: What is the maximum flow from s to t, other nodes $V - \{s,t\}$ can not produce / consume the commodity.

Flow conservation:

For $u \in V - \{s,t\}$: $\sum$ flow into u = $\sum$ flow out of u

$$= \sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$$

strategies for solving max flow

1. Augmenting path approaches.
   - start with ~~a~~ 0-flow
   - find paths from s to t on which flow can be increased
   - Reach max flow

   ★ At all times, we have a feasible flow.

2. Preflow - Push algorithms:
   - Flood the network with flow from s
   - what cannot reach t will flow back to s.

   (Idea: inspired by fluids)

   Feasible flow is obtained only at the end.

3. Scaling approach:

   Divide capacities and truncate (by 2)

   Recursively solve problem

   Scale up flows

   Adjust residual capacities.

Preflow-Push Relabel to front algorithm

(in Cormen's book)

1. Excess at each node u:

$$e(u) = \sum_{v \in V} f(v,u) - \sum_{v \in V} f(u,v) \geq 0$$

flow into u — flow out of u — for $u \in V - \{s\}$

$h(u)$ = height of u. $0 \leq h(u) \leq 2|V| - 2$

Initially: $\left.\begin{array}{l} h(s) = |V| \\ h(t) = 0 \end{array}\right\}$ unchangeable

$h(u) = 0$ for $u \in V - \{s,t\}$.

Push flow from u to v along (u,v):

$c_f(u,v) > 0$
- $(u,v) \in E$: $f(u,v) < c(u,v)$
- $(v,u) \in E$: $f(v,u) > 0$

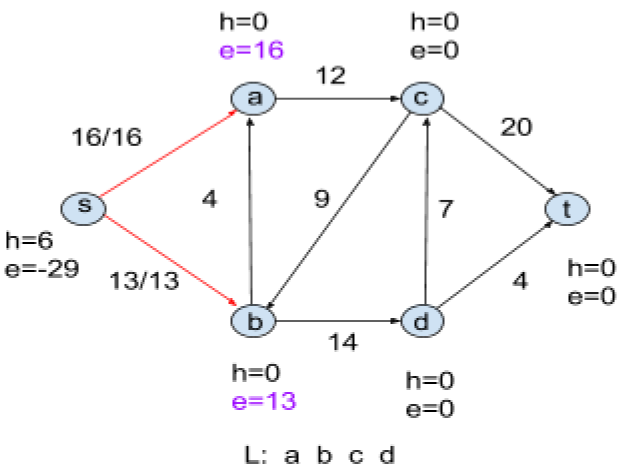
and $h(u) = h(v) + 1$

Relabel (u): $1 + \max\{h(v)\}$

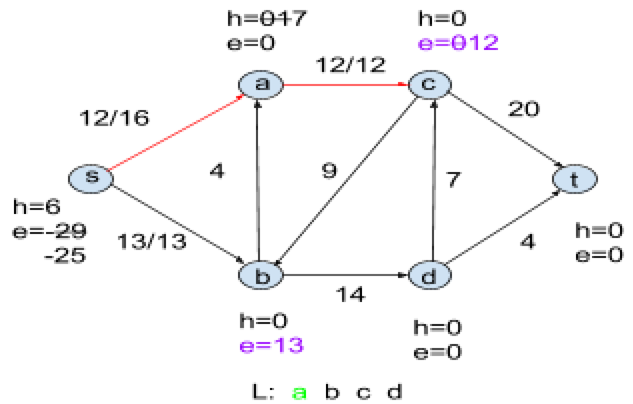
for all (u,v): $c_f(u,v) > 0$

<u>Flow example</u>: Preflow-push, relabel to front algorithm:

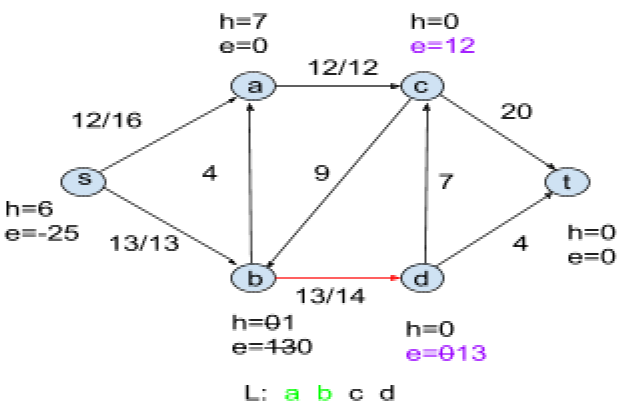
(1) Initial preflow:



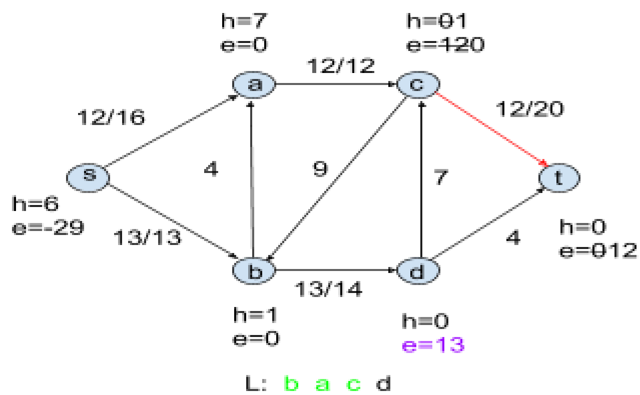
(2) Increase a.h to 1. Move a to front. Push excess flow 12 into (a,c). Raise a.h to 7. Move a to front. Push 4 units back to s.



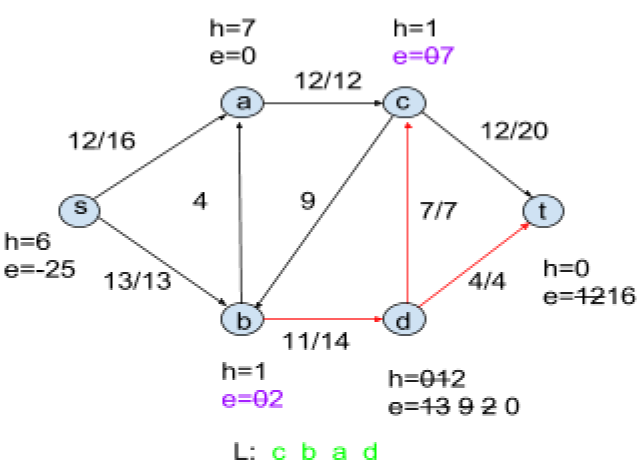
(3) Node a: no change. Raise b.h to 1. Move b to front. Push excess flow 13 into (b,d).



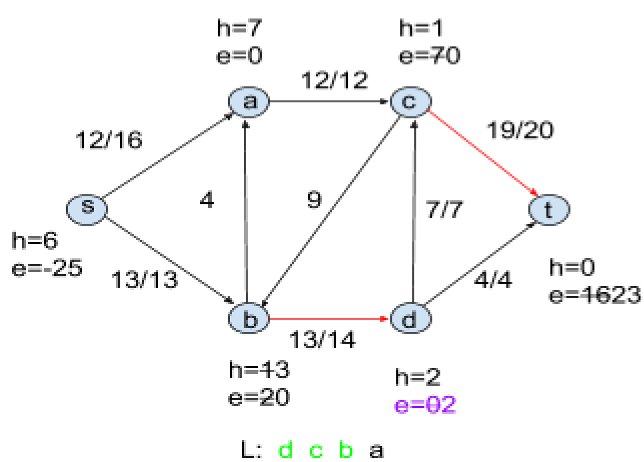
(4) Nodes b, a: no change. Raise c.h to 1. Move c to front. Push excess flow 12 into (c,t).



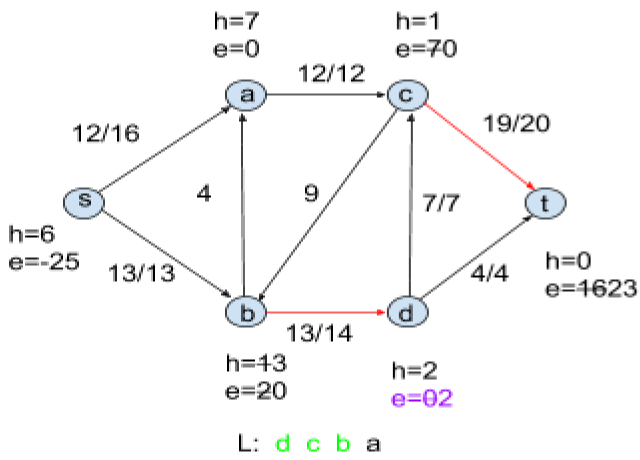
(5) Nodes c, b, a: no change. Increase d.h to 1. Move d to front. Push 4 units on (d,t). Raise d.h to 2. Push 7 units on (d,c). Push 2 units back to b.



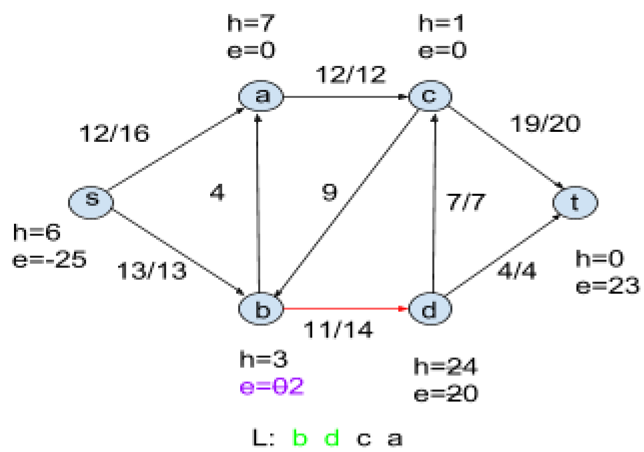
(6) Node d: no change. Node c: push excess flow 7 on (c,t). Raise b.h to 3. Move b to front. Push 2 units to d.



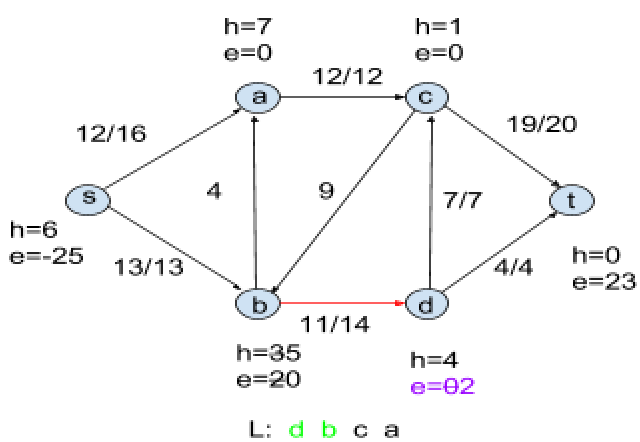
(6) Same figure shown again: b is moved to front.



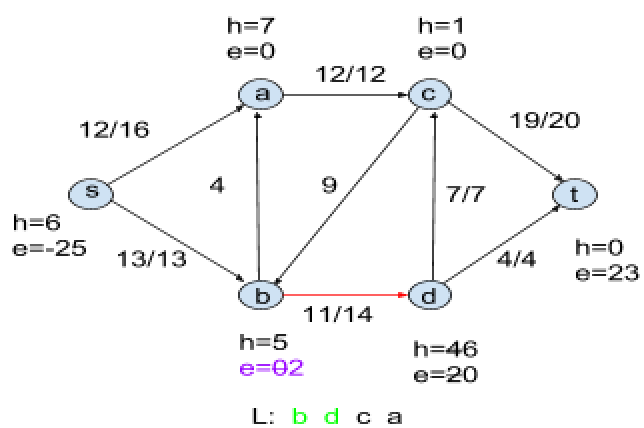
(7) Node b: no change. Raise d to 4. Move d to front. Push 2 units back on (b,d).



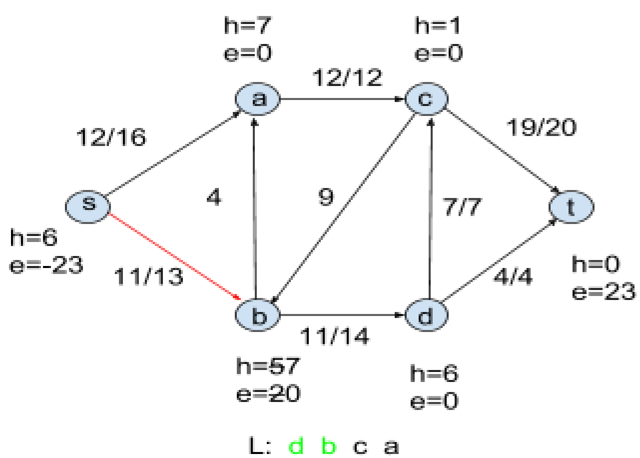
(8) Node d: no change. Raise b to 5. Move b to front. Push 2 units on (b,d).



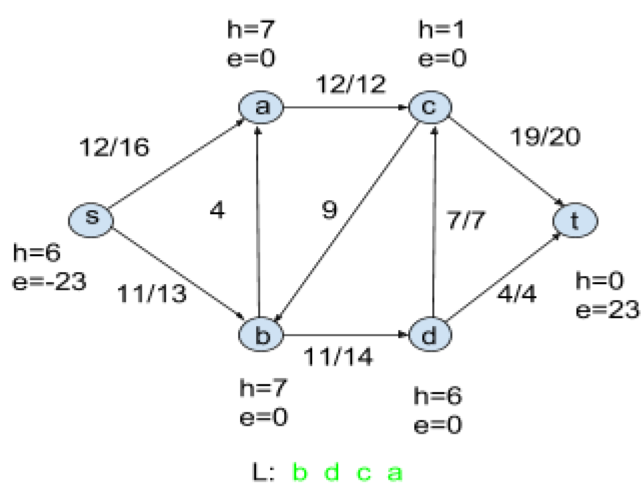
(9) Node b: no change. Raise d to 6. Move d to front. Push 2 units back on (b,d).



(10) Node d: no change. Raise b to 7. Move b to front. Push 2 units back on (s,b).

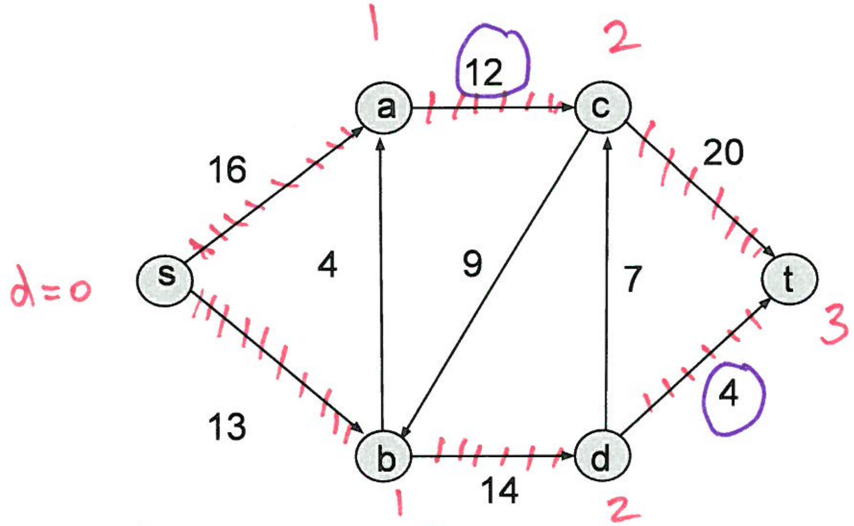


(11) No changes. Max flow is found.

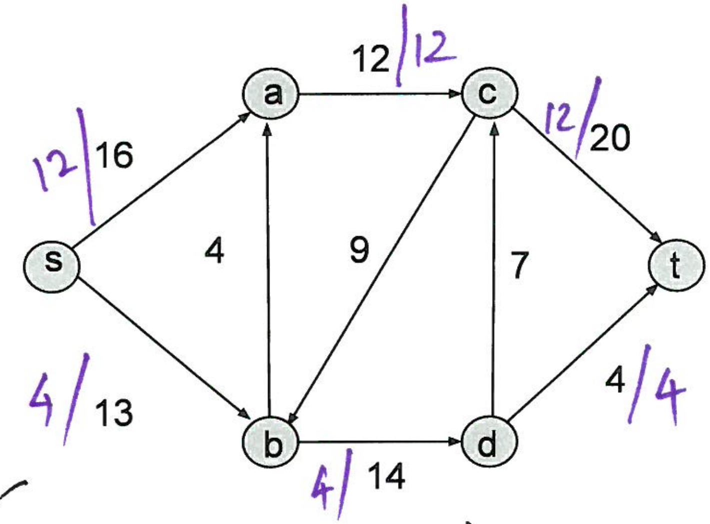


Dinitz algorithm – augmenting path approach

(improvement over Edmonds-Karp algorithm > Ford-Fulkerson's alg.)

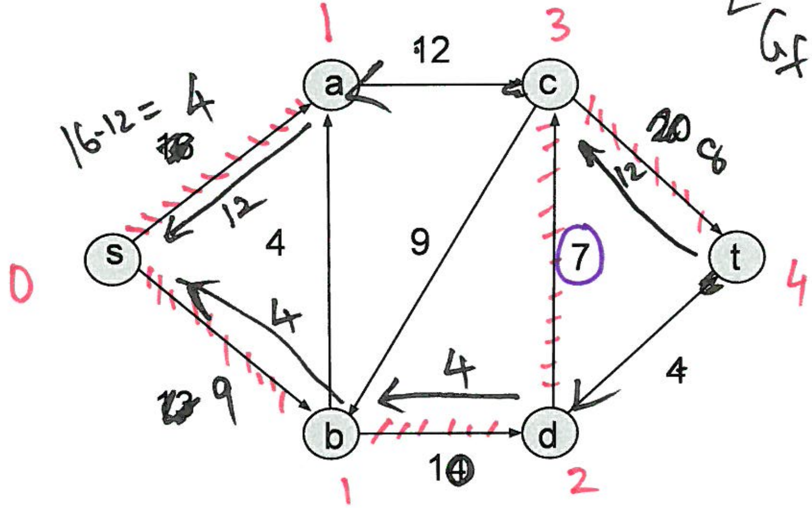


BFS(s): $G = G_f$



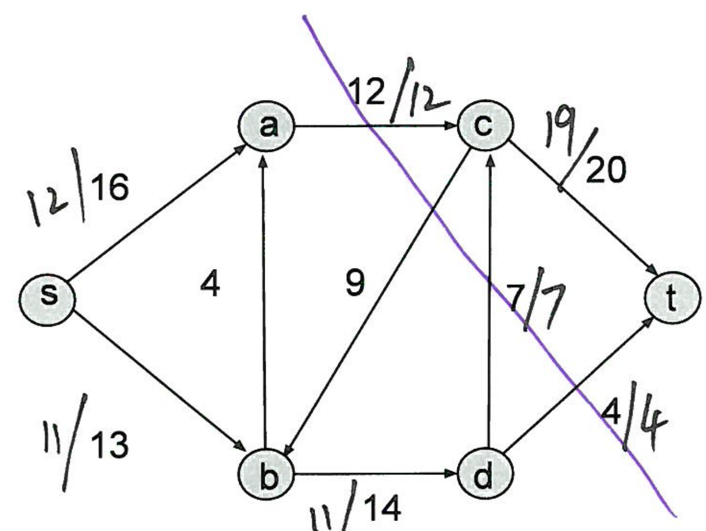
Flow of 16 units

G_f



BFS(s):

G_f of 16 flow.



G_f of 23 units