| | |
|---|---|
| CMSC 651 Advanced Algorithms | Lecture 11 |
| Lecturer: Samir Khuller | Th. Mar. 12, 1998 |

Original notes by Sibel Adali and Andrew Vakhutinsky.

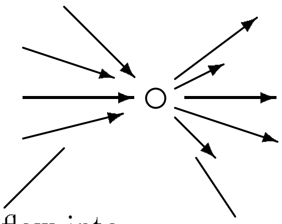
# 1 Network Flow - Maximum Flow Problem

Read [?, ?, ?].

The problem is defined as follows: Given a directed graph $G^d = (V, E, s, t, c)$ where $s$ and $t$ are special vertices called the source and the sink, and $c$ is a capacity function $c : E \rightarrow \Re^+$, find the maximum flow from $s$ to $t$.

Flow is a function $f : E \rightarrow \Re$ that has the following properties:

1. **(Skew Symmetry)** $f(u, v) = -f(v, u)$
2. **(Flow Conservation)** $\Sigma_{v \in V} f(u, v) = 0$ for all $u \in V - \{s, t\}$.
   (Incoming flow) $\Sigma_{v \in V} f(v, u) =$ (Outgoing flow) $\Sigma_{v \in V} f(u, v)$




Flow Conservation

3. **(Capacity Constraint)** $f(u, v) \leq c(u, v)$

Maximum flow is the maximum value $|f|$ given by

$$|f| = \Sigma_{v \in V} f(s, v) = \Sigma_{v \in V} f(v, t).$$

**Definition 1.1 (Residual Graph)** *$G_f^D$ is defined with respect to some flow function $f$, $G_f = (V, E_f, s, t, c')$ where $c'(u, v) = c(u, v) - f(u, v)$. Delete edges for which $c'(u, v) = 0$.*

As an example, if there is an edge in $G$ from $u$ to $v$ with capacity 15 and flow 6, then in $G_f$ there is an edge from $u$ to $v$ with capacity 9 (which means, I can still push 9 more units of liquid) and an edge from $v$ to $u$ with capacity 6 (which means, I can cancel all or part of the 6 units of liquid I'm currently pushing) [1]. $E_f$ contains all the edges $e$ such that $c'(e) > 0$.

**Lemma 1.2** *Here are some easy to prove facts:*

1. *$f'$ is a flow in $G_f$ iff $f + f'$ is a flow in $G$.*
2. *$f'$ is a maximum flow in $G_f$ iff $f + f'$ is a maximum flow in $G$.*
3. *$|f + f'| = |f| + |f'|$.*

[1] Since there was no edge from $v$ to $u$ in $G$, then its capacity was 0 and the flow on it was -6. Then, the capacity of this edge in $G_f$ is $0 - (-6) = 6$.

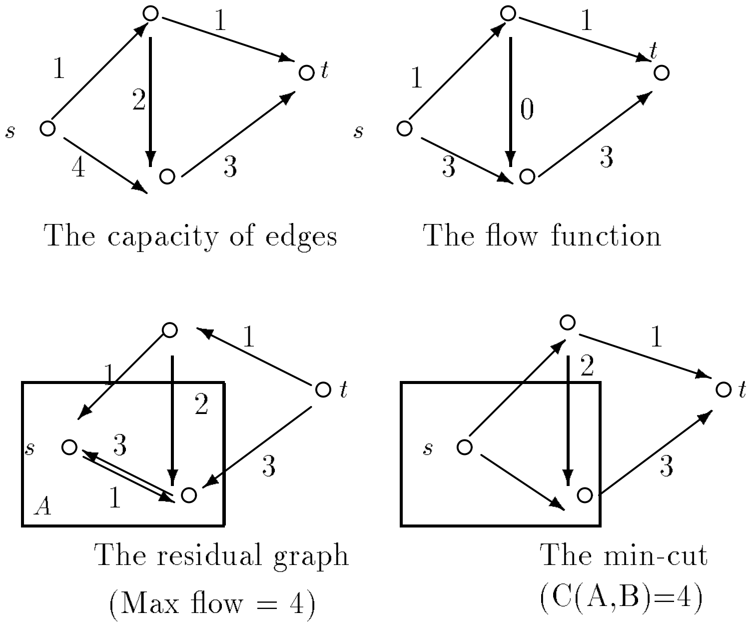



Figure 1: An example

*4. If $f$ is a flow in $G$, and $f^*$ is the maximum flow in $G$, then $f^* - f$ is the maximum flow in $G_f$.*

**Definition 1.3 (Augmenting Path)** *A path $P$ from $s$ to $t$ in the residual graph $G_f$ is called augmenting if for all edges $(u, v)$ on $P$, $c'(u, v) > 0$. The* residual capacity *of an augmenting path $P$ is $\min_{e \in P} c'(e)$.*

The idea behind this definition is that we can send a positive amount of flow along the augmenting path from $s$ to $t$ and "augment" the flow in $G$. (This flow increases the real flow on some edges and cancels flow on other edges, by reversing flow.)

**Definition 1.4 (Cut)** *An $(s, t)$ cut is a partitioning of $V$ into two sets $A$ and $B$ such that $A \cap B = \emptyset$, $A \cup B = V$ and $s \in A, t \in B$.*

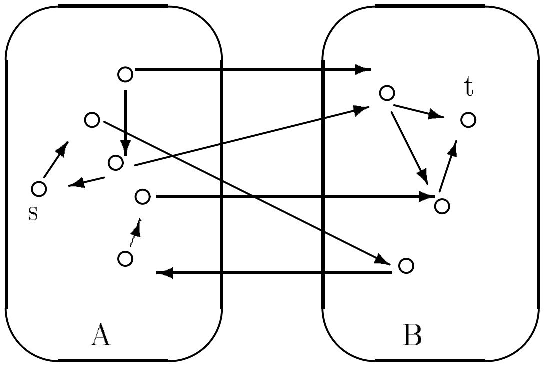



Figure 2: An $(s, t)$ Cut

**Definition 1.5 (Capacity Of A Cut)** *The capacity $C(A, B)$ is given by*

$$C(A, B) = \Sigma_{a \in A, b \in B} \ c(a, b).$$

By the capacity constraint we have that $|f| = \Sigma_{v \in V} f(s,v) \leq C(A,B)$ for any $(s,t)$ cut $(A,B)$. Thus the capacity of the minimum capacity $s,t$ cut is an upper bound on the value of the maximum flow.

**Theorem 1.6 (Max flow - Min cut Theorem)** *The following three statements are equivalent:*

1. *$f$ is a maximum flow.*
2. *There exists an $(s,t)$ cut $(A,B)$ with $C(A,B) = |f|$.*
3. *There are no augmenting paths in $G_f$.*

*An augmenting path is a directed path from $s$ to $t$.*

*Proof:*
We will prove that $(2) \Rightarrow (1) \Rightarrow (3) \Rightarrow (2)$.

**(**$(2) \Rightarrow (1)$**)** Since no flow can exceed the capacity of an $(s,t)$ cut (i.e. $f(A,B) \leq C(A,B)$), the flow that satisfies the equality condition of **(2)** must be the maximum flow.

**(**$(1) \Rightarrow (3)$**)** If there was an augmenting path, then I could augment the flow and find a larger flow, hence $f$ wouldn't be maximum.

**(**$(3) \Rightarrow (2)$**)** Consider the residual graph $G_f$. There is no directed path from $s$ to $t$ in $G_f$, since if there was this would be an augmenting path. Let $A = \{v | v \text{ is reachable from } s \text{ in } G_f\}$. $A$ and $\overline{A}$ form an $(s,t)$ cut, where all the edges go from $\overline{A}$ to $A$. The flow $f^{'}$ must be equal to the capacity of the edge, since for all $u \in A$ and $v \in \overline{A}$, the capacity of $(u,v)$ is 0 in $G_f$ and $0 = c(u,v) - f^{'}(u,v)$, therefore $c(u,v) = f^{'}(u,v)$. Then, the capacity of the cut in the original graph is the total capacity of the edges from $A$ to $\overline{A}$, and the flow is exactly equal to this amount. □

**A "Naive" Max Flow Algorithm:**

Initially let $f$ be the 0 flow
**while** (there is an augmenting path $P$ in $G_f$) **do**
    $c(P) \leftarrow \min_{e \in P} c'(e)$;
    send flow amount $c(P)$ along $P$;
    update flow value $|f| \leftarrow |f| + c(P)$;
    compute the new residual flow network $G_f$

Analysis: The algorithm starts with the zero flow, and stops when there are no augmenting paths from $s$ to $t$. If all edge capacities are integral, the algorithm will send at least one unit of flow in each iteration (since we only retain those edges for which $c'(e) > 0$). Hence the running time will be $O(m|f^*|)$ in the worst case ($|f^*|$ is the value of the max-flow).

**A worst case example.** Consider a flow graph as shown on the Fig. 3. Using augmenting paths $(s,a,b,t)$ and $(s,b,a,t)$ alternatively at odd and even iterations respectively (fig.1(b-c)), it requires total $|f^*|$ iterations to construct the max flow.

If all capacities are rational, there are examples for which the flow algorithm might never terminate. The example itself is intricate, but this is a fact worth knowing.

**Example.** Consider the graph on Fig. 4 where all edges except $(a,d)$, $(b,e)$ and $(c,f)$ are unbounded (have comparatively large capacities) and c$(a,d) = 1$, c$(b,e) = R$ and c$(c,f) = R^2$. Value $R$ is chosen such that $R = \frac{\sqrt{5}-1}{2}$ and, clearly (for any $n \geq 0$, $R^{n+2} = R^n - R^{n+1}$. If augmenting paths are selected as shown on Fig. 4 by dotted lines, residual capacities of the edges $(a,d)$, $(b,e)$ and $(c,f)$ will remain 0, $R^{3k+1}$ and $R^{3k+2}$ after every $(3k+1)$st iteration ($k = 0, 1, 2, \ldots$). Thus, the algorithm will never terminate.

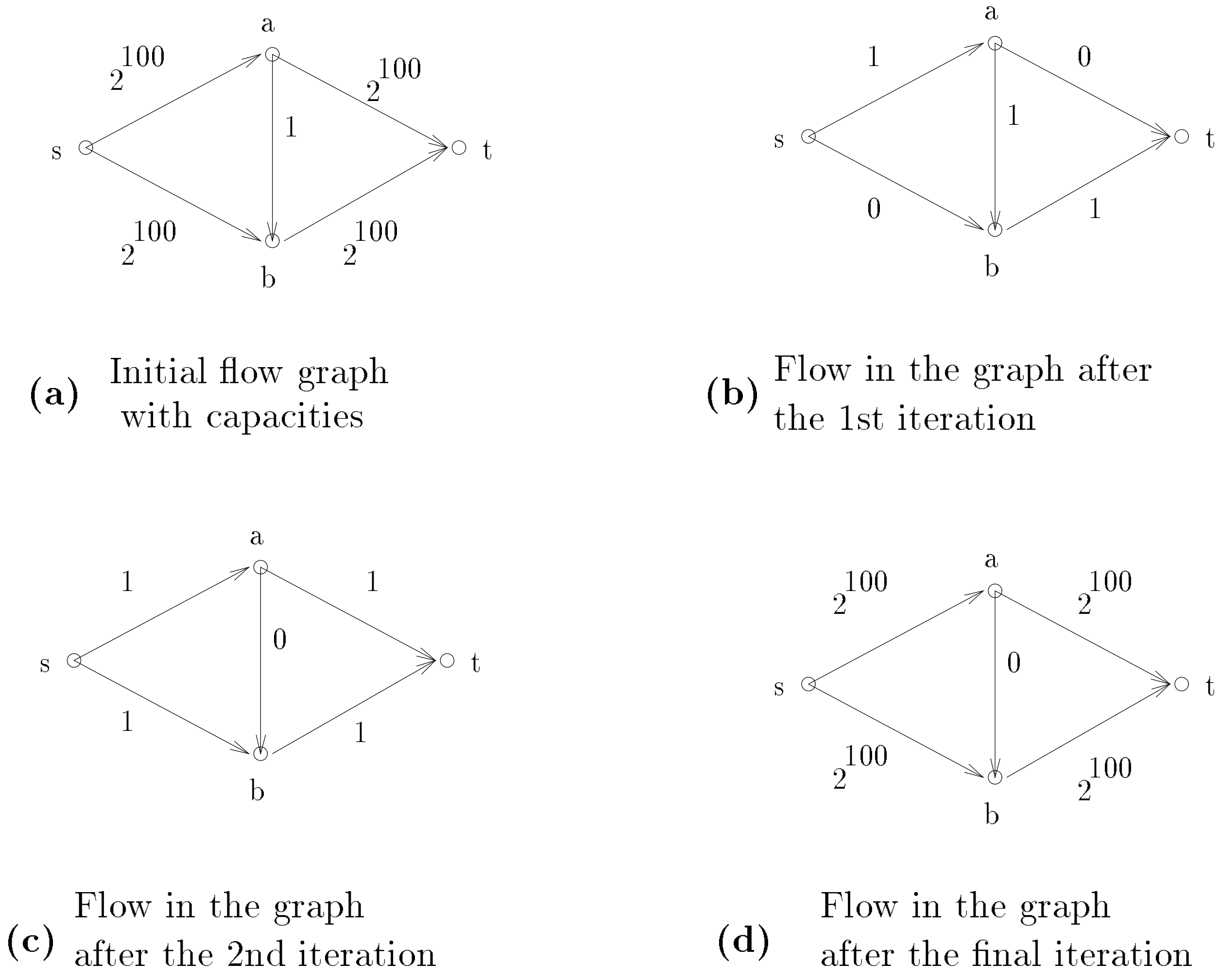



Figure 3: Worst Case Example

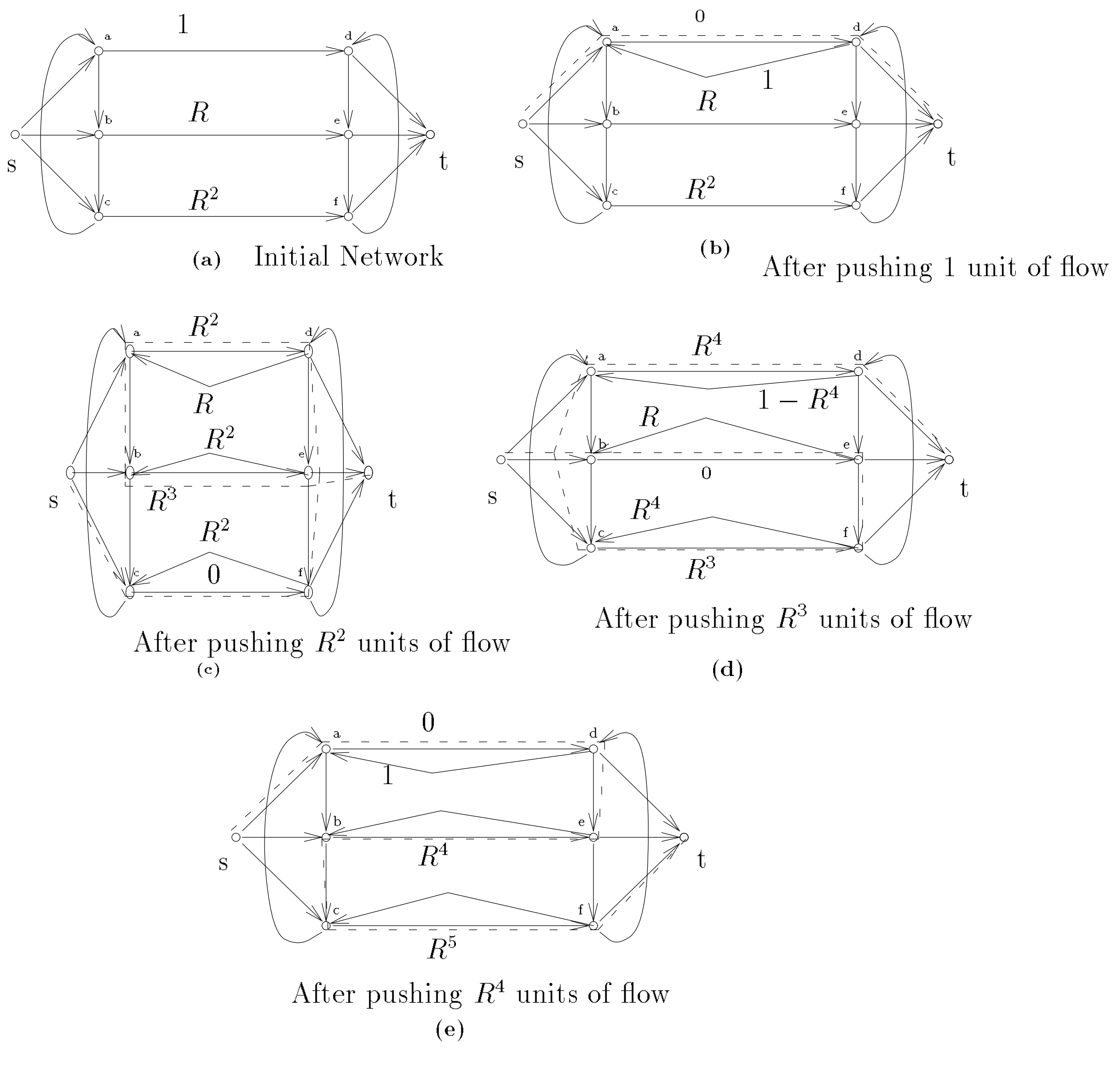



Figure 4: Non-terminating Example