Maximum Flow Problems IV.

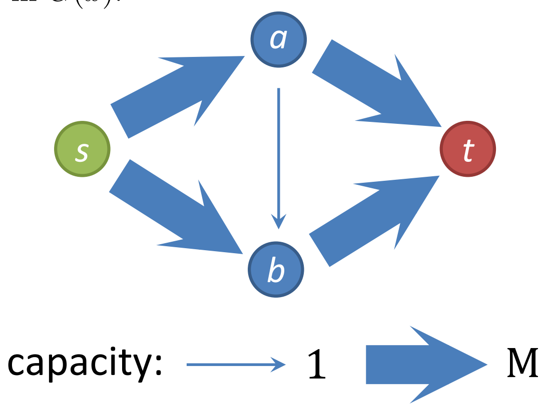
Review: $G = (V, E)$; edge capacity u; (s, t)-flow x; *Max-Flow Min-Cut Theorem*

1 Implementation and complexity

To find an x-augmenting path $\Rightarrow$ an *auxiliary graph* $G(x)$ defined as:

$$V(G(x)) = V \qquad E(G(x)) = \left\{ vw \;\middle|\; \text{or} \begin{array}{l} vw \in E \text{ and } x_{vw} < u_{vw} \\ wv \in E \text{ and } x_{wv} > 0 \end{array} \right\}$$

By definition, any st-path in $G(x)$ is x-augmenting (any path in $G(x)$ is x-increasing) Conversely, any x-augmenting path is an st-path in $G(x)$.



Augmenting Path algorithm:

1. Initialize $x = 0$
2. find an st-path P in $G(x)$;
3. if P exists $\Rightarrow$ augment x along P and go to 2.
4. else return x (a ***maximum flow***)

Possibly exponential number of iterations
. . . see the example on the right →

Starting with $x = 0$, alternately use augmenting paths s, a, b, t and s, b, a, t, both of x-width one $\Rightarrow 2 \times M$ iterations to reach maximum flow (M can be exponentially large)

Rule: (Edmonds-Karp) Always choose an augmenting path with smallest number of edges

Theorem 1. [Edmonds-Karp 1972] *If each augmentation uses a shortest augmenting path, then there are at most* $|V| \cdot |E|$ *augmentations before a maximum flow is found.*

Complexity of the *Augmenting Path* algorithm: $O(|V| \cdot |E|^2)$
– construction of an auxiliary graph $G(x)$ in $O(|E|)$
– finding an st-path in $G(x)$ with smallest # of edges $\Rightarrow$ by Breadth-First search $O(|E|)$
– augmenting x in $O(|V|)$ time (the path has always $< |V|$ edges)
– repeating $|V| \cdot |E|$ times

2 Sample Application

Transportation problem (special case: Maximum matching in bipartite graphs)

Theorem 2. *(Kőnig's Theorem) For a bipartite graph G, the maximum size of a* matching *equals the minimum size of a* vertex cover.

. . . consequence of the *Max-Flow Min-Cut Theorem*