26. Flow in Networks

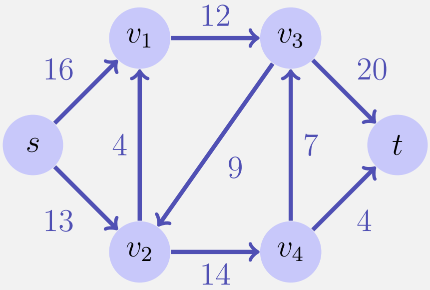
Flow Network, Maximal Flow, Cut, Rest Network, Max-flow Min-cut Theorem, Ford-Fulkerson Method, Edmonds-Karp Algorithm, Maximal Bipartite Matching [Ottman/Widmayer, Kap. 9.7, 9.8.1], [Cormen et al, Kap. 26.1-26.3]

Motivation

- Modelling flow of fluents, components on conveyors, current in electrical networks or information flow in communication networks.
- Connectivity of Communication Networks, Bipartite Matching, Circulation, Scheduling, Image Segmentation, Baseball Eliminination...

Flow Network

- *Flow network* $G = (V, E, c)$: directed graph with *capacities*
- Antiparallel edges forbidden: $(u, v) \in E \;\Rightarrow\; (v, u) \notin E$.
- Model a missing edge (u, v) by $c(u, v) = 0$.
- *Source* s and *sink* t: special nodes. Every node v is on a path between s and t : $s \rightsquigarrow v \rightsquigarrow t$

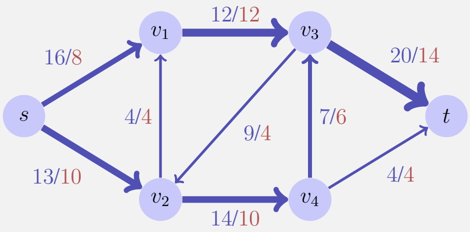


Flow

A *Flow* $f : V \times V \to \mathbb{R}$ fulfills the following conditions:

- *Bounded Capacity*:
  For all $u, v \in V$: $f(u,v) \leq c(u,v)$.
- *Skew Symmetry*:
  For all $u, v \in V$: $f(u,v) = -f(v,u)$.
- *Conservation of flow*:
  For all $u \in V \setminus \{s,t\}$:

$$\sum_{v \in V} f(u,v) = 0.$$



Value of the flow:
$|f| = \sum_{v \in V} f(s,v)$.
Here $|f| = 18$.

How large can a flow possibly be?

Limiting factors: cuts

- *cut separating s from t*: Partition of V into S and T with $s \in S$, $t \in T$.
- *Capacity* of a cut: $c(S,T) = \sum_{v \in S, v' \in T} c(v, v')$
- *Minimal cut*: cut with minimal capacity.
- *Flow over the cut*: $f(S,T) = \sum_{v \in S, v' \in T} f(v, v')$

Implicit Summation

Notation: Let $U, U' \subseteq V$

$$f(U,U') := \sum_{\substack{u\in U\\ u'\in U'}} f(u,u'), \qquad f(u,U') := f(\{u\},U')$$

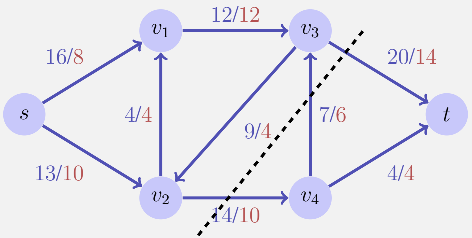
Thus

- $|f| = f(s,V)$
- $f(U,U) = 0$
- $f(U,U') = -f(U',U)$
- $f(X\cup Y,Z) = f(X,Z) + f(Y,Z)$, if $X\cap Y = \emptyset$.
- $f(R,V) = 0$ if $R\cap\{s,t\} = \emptyset$. [flow conversation!]

How large can a flow possibly be?

For each flow and each cut it holds that $f(S,T) = |f|$:

$$f(S,T) = f(S,V) - \underbrace{f(S,S)}_{0} = f(S,V)$$
$$= f(s,V) + f(\underbrace{S - \{s\}}_{\not\ni t, \not\ni s}, V) = |f|.$$

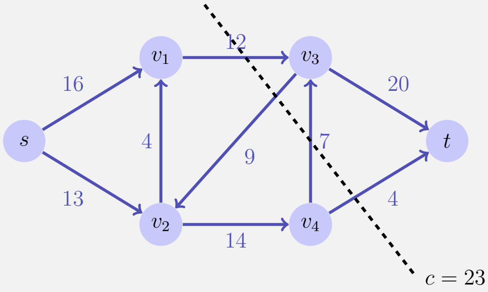


Maximal Flow ?

In particular, for each cut (S, T) of V.

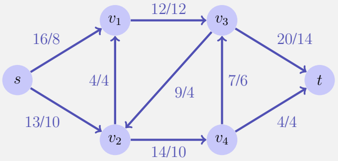
$$|f| \leq \sum_{v \in S, v' \in T} c(v, v') = c(S, T)$$

Will discover that equality holds for $\min_{S,T} c(S, T)$.



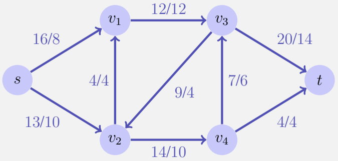
Maximal Flow ?

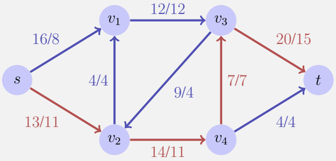
Naive Procedure



Maximal Flow ?

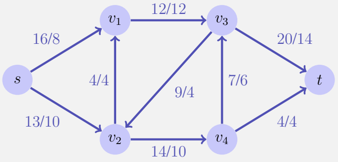
Naive Procedure

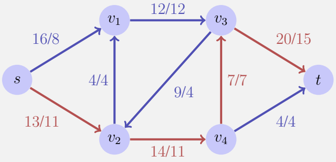


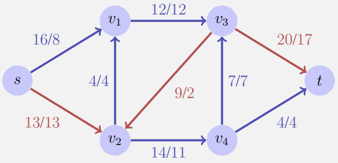


Maximal Flow ?

Naive Procedure

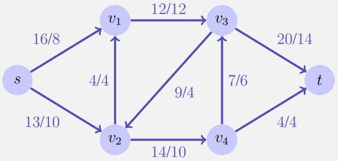


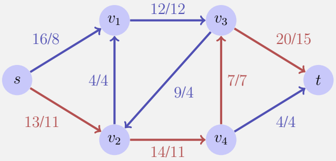


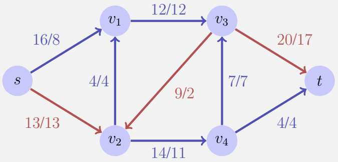


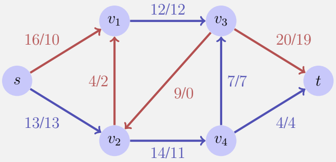
Maximal Flow ?

Naive Procedure



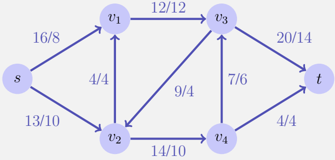


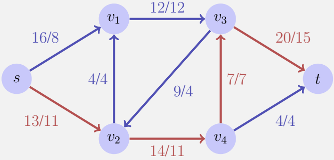


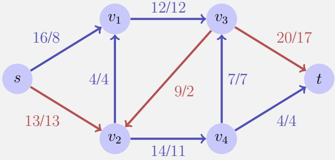


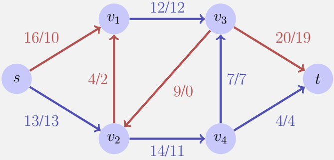
Maximal Flow ?

Naive Procedure









Conclusion: greedy increase of flow does not solve the problem.

The Method of Ford-Fulkerson

- Start with $f(u, v) = 0$ for all $u, v \in V$
- Determine rest network* G_f and expansion path in G_f
- Increase flow via expansion path*
- Repeat until no expansion path available.

$$
\begin{aligned}
G_f &:= (V, E_f, c_f) \\
c_f(u, v) &:= c(u, v) - f(u, v) \quad \forall u, v \in V \\
E_f &:= \{(u, v) \in V \times V | c_f(u, v) > 0\}
\end{aligned}
$$

*Will now be explained

Increase of flow, negative!

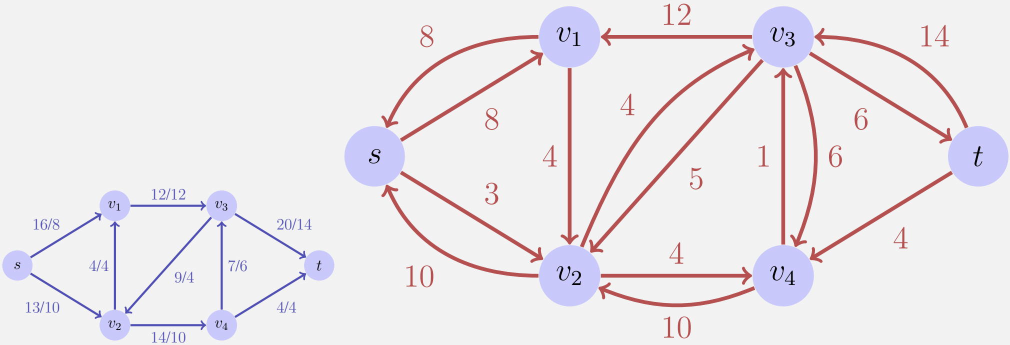
Let some flow f in the network be given.

Finding:

- Increase of the flow along some edge possible, when flow can be increased along the edge,i.e. if $f(u, v) < c(u, v)$.
  Rest capacity $c_f(u, v) = c(u, v) - f(u, v) > 0$.
- Increase of flow *against the direction* of the edge possible, if flow can be reduced along the edge, i.e. if $f(u, v) > 0$.
  Rest capacity $c_f(v, u) = f(u, v) > 0$.

Rest Network

Rest network G_f provided by the edges with positive rest capacity:



Rest networks provide the same kind of properties as flow networks with the exception of permitting antiparallel capacity-edges

Observation

Theorem

Let $G = (V, E, c)$ be a flow network with source s and sink t and f a flow in G. Let G_f be the corresponding rest networks and let f' be a flow in G_f. Then $f \oplus f'$ with

$$(f \oplus f')(u, v) = f(u, v) + f'(u, v)$$

defines a flow in G with value $|f| + |f'|$.

Proof

$f \oplus f'$ defines a flow in G:

- capacity limit

$$(f \oplus f')(u,v) = f(u,v) + \underbrace{f'(u,v)}_{\le c(u,v) - f(u,v)} \le c(u,v)$$

- skew symmetry

$$(f \oplus f')(u,v) = -f(v,u) + -f'(v,u) = -(f \oplus f')(v,u)$$

- flow conservation $u \in V - \{s,t\}$:

$$\sum_{v \in V} (f \oplus f')(u,v) = \sum_{v \in V} f(u,v) + \sum_{v \in V} f'(u,v) = 0$$

Proof

Value of $f \oplus f'$

$$
\begin{aligned}
|f \oplus f'| &= (f \oplus f')(s, V) \\
&= \sum_{u \in V} f(s, u) + f'(s, u) \\
&= f(s, V) + f'(s, V) \\
&= |f| + |f'|
\end{aligned}
$$

$\blacksquare$

Augmenting Paths

expansion path p: simple path from s to t in the rest network G_f.

Rest capacity $c_f(p) = \min\{c_f(u,v) : (u,v)$ edge in $p\}$

Flow in G_f

Theorem

The mapping $f_p : V \times V \to \mathbb{R}$,

$$f_p(u,v) = \begin{cases} c_f(p) & \text{if } (u,v) \text{ edge in } p \\ -c_f(p) & \text{if } (v,u) \text{ edge in } p \\ 0 & \text{otherwise} \end{cases}$$

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

f_p is a flow (easy to show). there is one and only one $u \in V$ with $(s,u) \in p$. Thus $|f_p| = \sum_{v \in V} f_p(s,v) = f_p(s,u) = c_f(p)$.

Consequence

Strategy for an algorithm:

With an expansion path p in G_f the flow $f \oplus f_p$ defines a new flow with value $|f \oplus f_p| = |f| + |f_p| > |f|$.

Max-Flow Min-Cut Theorem

Theorem

Let f be a flow in a flow network $G = (V, E, c)$ with source s and sink t. The following statementsa are equivalent:

1. *f is a maximal flow in G*
2. *The rest network G_f does not provide any expansion paths*
3. *It holds that $|f| = c(S, T)$ for a cut (S, T) of G.*

Proof

- $(3) \Rightarrow (1)$:
  It holds that $|f| \leq c(S,T)$ for all cuts S,T. From $|f| = c(S,T)$ it follows that $|f|$ is maximal.
- $(1) \Rightarrow (2)$:
  f maximal Flow in G. Assumption: G_f has some expansion path $|f \oplus f_p| = |f| + |f_p| > |f|$. Contradiction.

Proof $(2) \Rightarrow (3)$

Assumption: G_f has no expansion path

Define $S = \{v \in V : \text{ there is a path } s \rightsquigarrow v \text{ in } G_f\}$.

$(S, T) := (S, V \setminus S)$ is a cut: $s \in S, t \in T$.

Let $u \in S$ and $v \in T$. Then $c_f(u, v) = 0$, also
$c_f(u, v) = c(u, v) - f(u, v) = 0$. Somit $f(u, v) = c(u, v)$.

Thus

$$|f| = f(S, T) = \sum_{u \in S} \sum_{v \in T} f(u, v) = \sum_{u \in S} \sum_{v \in T} c(u, v) = C(S, T).$$

$\blacksquare$

Algorithm Ford-Fulkerson(G, s, t)

Input: Flow network $G = (V, E, c)$
Output: Maximal flow f.

for $(u, v) \in E$ **do**
    $f(u, v) \leftarrow 0$

while Exists path $p : s \rightsquigarrow t$ in rest network G_f **do**
    $c_f(p) \leftarrow \min\{c_f(u, v) : (u, v) \in p\}$
    foreach $(u, v) \in p$ **do**
        $f(u, v) \leftarrow f(u, v) + c_f(p)$
        $f(v, u) \leftarrow f(v, u) - c_f(p)$

Practical Consideration

In an implementation of the Ford-Fulkerson algorithm the negative flow egdes are usually not stored because their value always equals the negated value of the antiparallel edge.

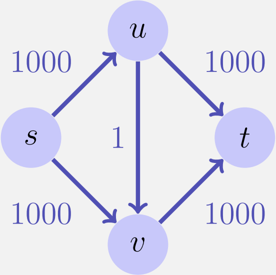
$$f(u,v) \leftarrow f(u,v) + c_f(p)$$
$$f(v,u) \leftarrow f(v,u) - c_f(p)$$

is then transformed to

if $(u,v) \in E$ **then**
  $f(u,v) \leftarrow f(u,v) + c_f(p)$
else
  $f(v,u) \leftarrow f(v,u) - c_f(p)$

Analysis

- The Ford-Fulkerson algorithm does not necessarily have to converge for irrational capacities. For integers or rational numbers it terminates.
- For an integer flow, the algorithms requires maximally $|f_{\max}|$ iterations of the while loop (because the flow increases minimally by 1). Search a single increasing path (e.g. with DFS or BFS) $\mathcal{O}(|E|)$ Therefore $\mathcal{O}(f_{\max}|E|)$.



With an unlucky choice the algorithm may require up to 2000 iterations here.

Edmonds-Karp Algorithm

Choose in the Ford-Fulkerson-Method for finding a path in G_f the expansion path of shortest possible length (e.g. with BFS)

Edmonds-Karp Algorithm

Theorem

When the Edmonds-Karp algorithm is applied to some integer valued flow network $G = (V, E)$ with source s and sink t then the number of flow increases applied by the algorithm is in $\mathcal{O}(|V| \cdot |E|)$.
$\Rightarrow$ *Overal asymptotic runtime:* $\mathcal{O}(|V| \cdot |E|^2)$

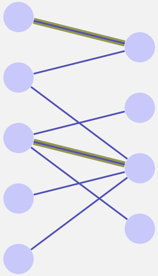
[Without proof]

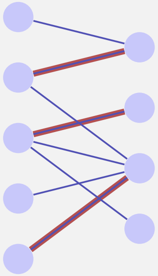
Application: maximal bipartite matching

Given: bipartite undirected graph $G = (V, E)$.

Matching M: $M \subseteq E$ such that $|\{m \in M : v \in m\}| \leq 1$ for all $v \in V$.

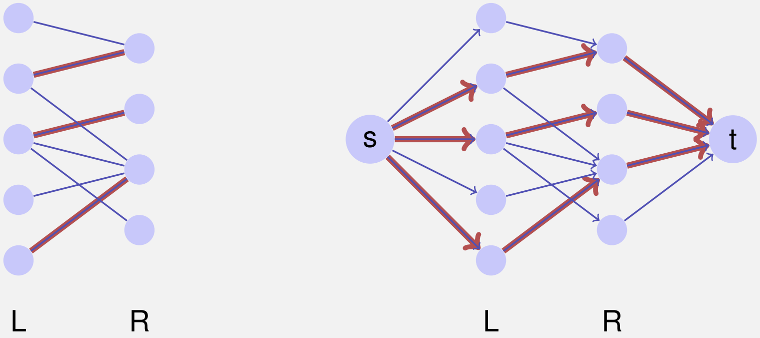
Maximal Matching M: Matching M, such that $|M| \geq |M'|$ for each matching M'.





Corresponding flow network

Construct a flow network that corresponds to the partition L, R of a bipartite graph with source s and sink t, with directed edges from s to L, from L to R and from R to t. Each edge has capacity 1.



Integer number theorem

Theorem

If the capacities of a flow network are integers, then the maximal flow generated by the Ford-Fulkerson method provides integer numbers for each $f(u, v)$*,* $u, v \in V$*.*

[without proof]

Consequence: Ford-Fulkerson generates for a flow network that corresponds to a bipartite graph a maximal matching $M = \{(u, v) : f(u, v) = 1\}$.