\[ S^+ = SUT - (S \wedge T) \]

\[ \Rightarrow M \oplus P \text{ is a matching with } |M \oplus P| \geq |M| + 1. \]

**Alg:**

\[
\begin{align*}
M &= \emptyset \\
\text{repeat} \{ & \text{find augmenting path } P \text{ for } M \\
& \text{if not exist, return } M \\
& M = M \oplus P
\}
\]

**Thm:**

matching \( M \) is optimal \( \Rightarrow \) it has no augmenting path.

- each iteration: \( O(m) \) time
- \# iterations: \( \leq n \)

\[ \Rightarrow O(mn) \]

**Lemma**

If \( M \) and \( M^* \) are matchings with \( |M| < |M^*| \), then \( M \oplus M^* \) contains \( |M^*| - |M| \) vertex-disjoint augmenting paths for \( M \):

**Pf:** \( M \oplus M^* \) has max deg \( \leq 2 \). \( M^* \)

**Fact**

Any undir. graph with max deg \( \leq 2 \) can be decomposed into vertex-disjoint paths & cycles.
$M \cup M^*$

Cycle  even path  blue-odd path  red-odd path

$|M^* - |M| = \# \text{ blue-odd paths} - \# \text{ red-odd paths}$

$\leq \# \text{ blue-odd paths}$  
underlined: these are aug paths

Proof of Thm:

If $\exists$ any path, then $M$ is not optimal.

If $M$ is not optimal,

by Lemma, $\exists > 1$ aug path.  \[ \square \]

How to find an augmenting path?

form directed graph $G_M$: $u, v$ on left side,  
if $uv \in E - M$, add $(u,v)$ to $G_M$.  
if $uv \in M$, add $(v,u)$ to $G_M$.  
if $u$ not covered by $M$, add $(s,u)$ to $G_M$.  
if $v$ not covered by $M$, add $(v,t)$ to $G_M$.  
$O(m)$ time
if u not covered by M, add (u, t) to G_M
Run BFS/DFS on a directed graph G_M from s to t

M = \{2B, 4D\}

# iterations ≤ |M*| since |M| increases by 1 per iteration

⇒ O(mn) time

Hopcroft & Karp's Alg '73: Sketch

Idea 1 - always pick shortest augmenting path
Idea 2 - augment multiple vertex-disjoint shortest paths per iteration.

Fact: length of shortest paths increases per iteration

⇒ after V_n iterations, path lengths ≥ V_n
⇒ ≤ V_n vertex-disj. aug paths
⇒ by lemma, \(|M^*| - |M| \) ≤ V_n.
⇒ |M| ≥ |M^*| - V_n
⇒ # remaining iterations ≤ V_n.
⇒ O(mV_n) time
Extensions: unweighted general (non-bipartite)
Edmonds '65 \(O(mn^2)\)
'80 \(O(mVn)\)

weighted bipartite
weighted non-bipartite

Max Flow

Problem: Given directed graph \(G = (V, E), s, t \in V\)
\(|V| = n, |E| = m,\)
where each \(e \in E\) has a capacity \(c(e) > 0,\)
find flow from \(s\) to \(t\) with max value.

Def: A flow is \(f: E \rightarrow \mathbb{R}\) s.t.
\(f(u, v) \in E,\quad 0 \leq f(u, v) \leq c(u, v)\)
$u \in V - \{s,t\}$: 

$\sum_{u: (u,v) \in E} f(u,v) = \sum_{w: (v,w) \in E} f(v,w)$

Value of $f = ||f|| = \sum_{v: (s,v) \in E} f(s,v)

(= \sum_{v: (v,t) \in E} f(v,t))$

(Convention: make $f(v,u) = -f(u,v)$ if $(u,v) \in E$ and $f(u,v) = 0$ if $(u,v), (v,u) \notin E$.)

then flow conservation means $\sum_{w\in V} f(v,w) = 0$.

Apply! Special Case 1: max bipartite matching

Diagram:

```
\begin{tikzpicture}
    \node (s) at (0,0) [circle, draw] {s};
    \node (v1) at (1,1) [circle, draw] {v1};
    \node (v2) at (1,0) [circle, draw] {v2};
    \node (v3) at (1,-1) [circle, draw] {v3};
    \node (v4) at (2,0) [circle, draw] {v4};
    \node (v5) at (3,0) [circle, draw] {v5};
    \node (v6) at (2,1) [circle, draw] {v6};
    \node (u1) at (0,-2) [circle, draw] {u1};
    \node (u2) at (0,-1) [circle, draw] {u2};
    \node (u3) at (0,1) [circle, draw] {u3};
    \node (u4) at (1,-2) [circle, draw] {u4};
    \node (u5) at (1,-1) [circle, draw] {u5};
    \node (u6) at (1,1) [circle, draw] {u6};

    \draw[->] (s) -- (v1);
    \draw[->] (s) -- (v2);
    \draw[->] (s) -- (v3);
    \draw[->] (s) -- (v4);
    \draw[->] (s) -- (v5);
    \draw[->] (s) -- (v6);
    \draw[->] (v1) -- (u1);
    \draw[->] (v1) -- (u2);
    \draw[->] (v1) -- (u3);
    \draw[->] (v2) -- (u4);
    \draw[->] (v2) -- (u5);
    \draw[->] (v2) -- (u6);
    \draw[->] (v3) -- (u1);
    \draw[->] (v3) -- (u2);
    \draw[->] (v3) -- (u3);
    \draw[->] (v4) -- (u4);
    \draw[->] (v4) -- (u5);
    \draw[->] (v4) -- (u6);
    \draw[->] (v5) -- (u1);
    \draw[->] (v5) -- (u2);
    \draw[->] (v5) -- (u3);
    \draw[->] (v6) -- (u4);
    \draw[->] (v6) -- (u5);
    \draw[->] (v6) -- (u6);
\end{tikzpicture}
```