Lecture 5: Matchings

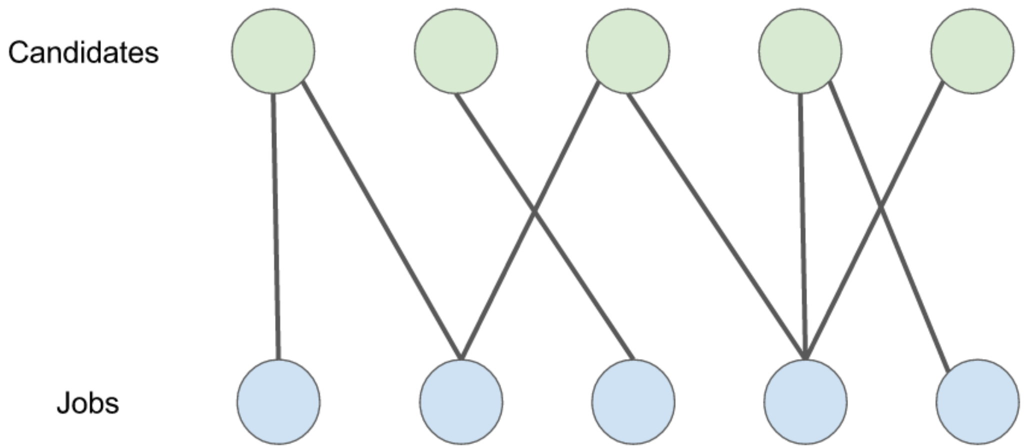
Shuai Li

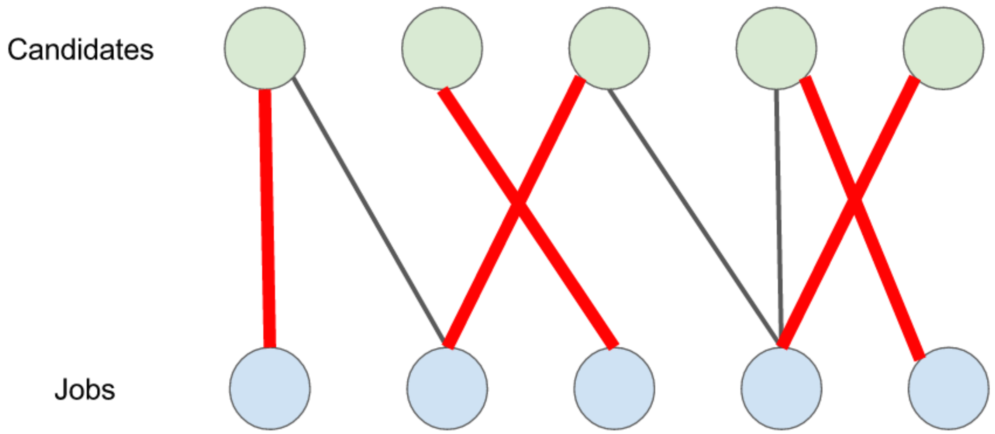
John Hopcroft Center, Shanghai Jiao Tong University

https://shuaili8.github.io

https://shuaili8.github.io/Teaching/CS445/index.html

Motivating example



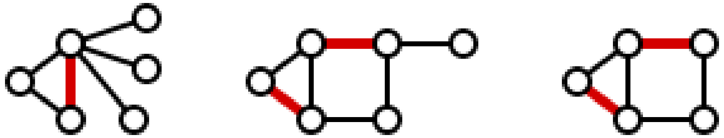


Definitions

- A matching is a set of independent edges, in which no pair shares a vertex
- The vertices incident to the edges of a matching M are M-saturated; the others are M-unsaturated
- A perfect matching in a graph is a matching that saturates every vertex
- Example (3.1.2, W) The number of perfect matchings in $K_{n,n}$ is $n!$
- Example (3.1.3, W) The number of perfect matchings in K_{2n} is
$$f_n = (2n-1)(2n-3)\cdots 1 = (2n-1)!!$$

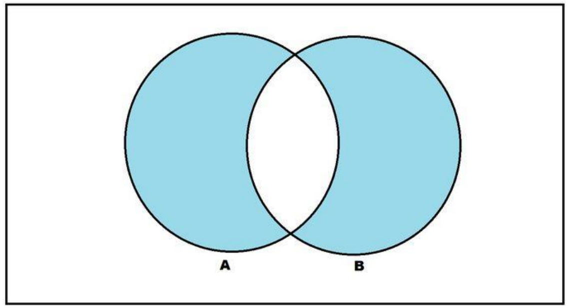
Maximal/maximum matchings 极大/最大

- A maximal matching in a graph is a matching that cannot be enlarged by adding an edge
- A maximum matching is a matching of maximum size among all matchings in the graph
- Example: P_3, P_5

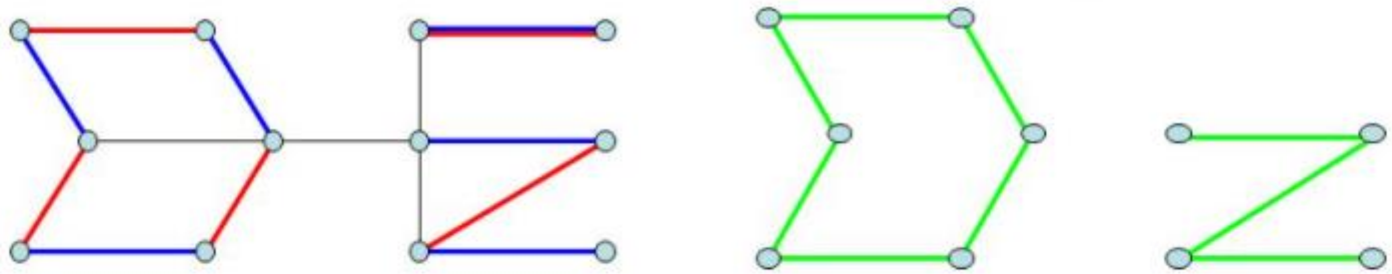


- Every maximum matching is maximal, but not every maximal matching is a maximum matching

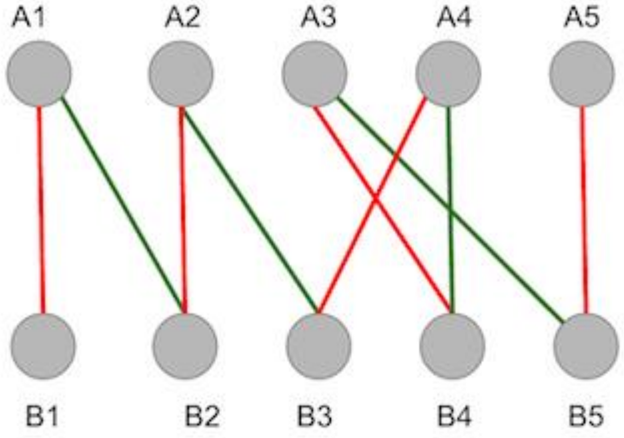
Symmetric difference of matchings



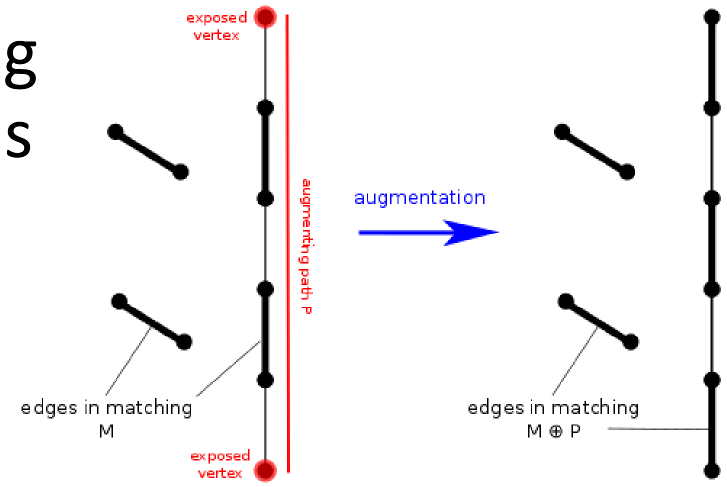
- The symmetric difference of M, M' is $M \Delta M' = (M - M') \cup (M' - M)$
- Lemma (3.1.9, W) Every component of the symmetric difference of two matchings is a path or an even cycle



Maximum matching and augmenting path



- Given a matching M, an M-alternating path is a path that alternates between edges in M and edges not in M
- An M-alternating path whose endpoints are M-unsaturated is an M-augmenting path
- Theorem (3.1.10, W; 1.50, H; Berge 1957) A matching M in a graph G is a maximum matching in $G \Leftrightarrow G$ has no M-augmenting path



Hall's theorem (TONCAS)

- Theorem (3.1.11, W; 1.51, H; 2.1.2, D; Hall 1935) Let G be a bipartite graph with partition X, Y.
  G contains a matching of $X \Leftrightarrow |N(S)| \geq |S|$ for all $S \subseteq X$

> Theorem (3.1.10, W; 1.50, H; Berge 1957) A matching M in a graph G is a maximum matching in $G \Leftrightarrow G$ has no M-augmenting path

- Exercise. Read the other two proofs in Diestel.
- Corollary (3.1.13, W; 2.1.3, D) Every k-regular ($k > 0$) bipartite graph has a perfect matching

General regular graph

- Corollary (2.1.5, D) Every regular graph of positive even degree has a 2-factor
  - A k-regular spanning subgraph is called a k-factor
  - A perfect matching is a 1-factor

Application to SDR

- Given some family of sets X, a system of distinct representatives for the sets in X is a 'representative' collection of distinct elements from the sets of X

$$S_1 = \{2, 8\},$$
$$S_2 = \{8\},$$
$$S_3 = \{5, 7\},$$
$$S_4 = \{2, 4, 8\},$$
$$S_5 = \{2, 4\}.$$

The family $X_1 = \{S_1, S_2, S_3, S_4\}$ does have an SDR, namely $\{2, 8, 7, 4\}$. The family $X_2 = \{S_1, S_2, S_4, S_5\}$ does not have an SDR.

- Theorem(1.52, H) Let $S_1, S_2, \dots, S_k$ be a collection of finite, nonempty sets. This collection has SDR $\Leftrightarrow$ for every $t \in [k]$, the union of any t of these sets contains at least t elements

König Theorem
Augmenting Path Algorithm

Vertex cover

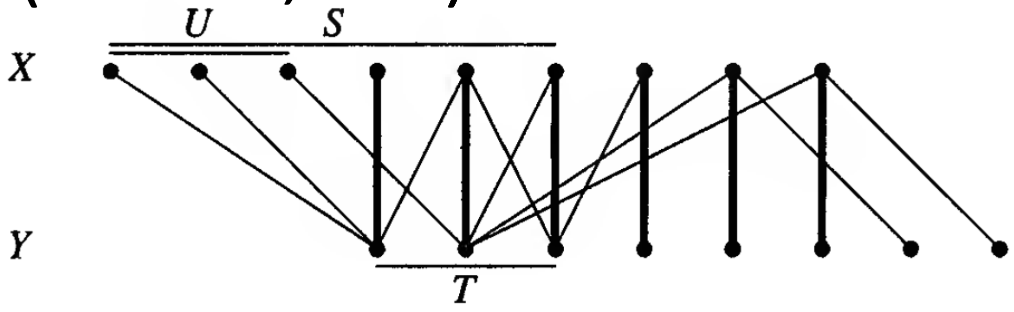
- A set $U \subseteq V$ is a (vertex) cover of E if every edge in G is incident with a vertex in U
- Example:
  - Art museum is a graph with hallways are edges and corners are nodes
  - A security camera at the corner will guard the paintings on the hallways
  - The minimum set to place the cameras?

König-Egeváry Theorem (Min-max theorem)

- Theorem (3.1.16, W; 1.53, H; 2.1.1, D; König 1931; Egeváry 1931) Let G be a bipartite graph. The maximum size of a matching in G is equal to the minimum size of a vertex cover of its edges

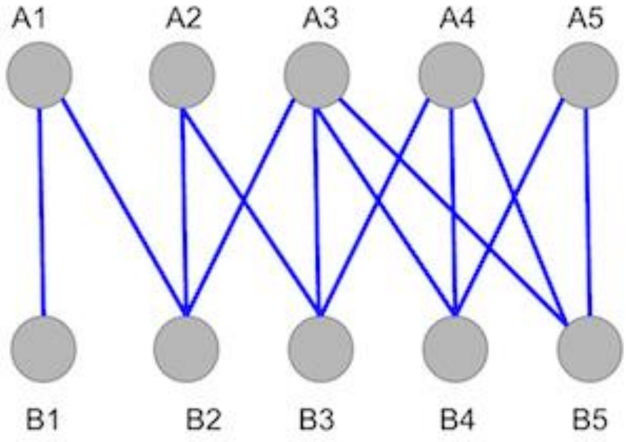
Theorem (3.1.10, W; 1.50, H; Berge 1957) A matching M in a graph G is a maximum matching in $G \Leftrightarrow G$ has no M-augmenting path

Augmenting path algorithm (3.2.1, W)

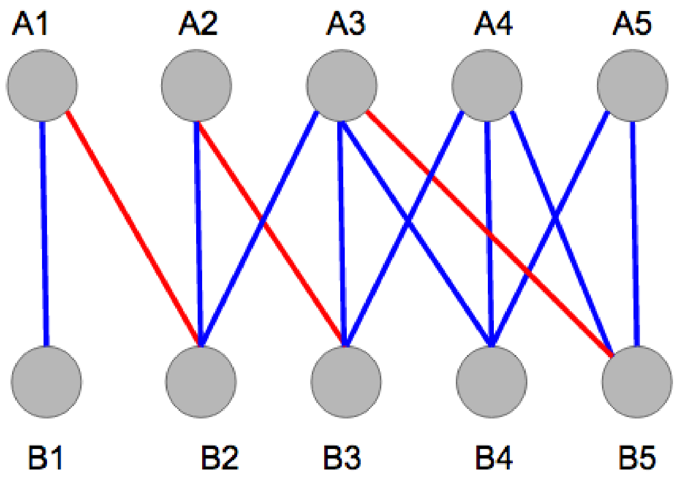


- **Input**: $G = B(X, Y)$, a matching M in G
  $U = \{M\text{–unsaturated vertices in } X\}$
- **Idea**: Explore M-alternating paths from U letting $S \subseteq X$ and $T \subseteq Y$ be the sets of vertices reached
- **Initialization**: $S = U, T = \emptyset$ and all vertices in S are unmarked
- **Iteration**:
  - If S has no unmarked vertex, stop and report $T \cup (X - S)$ as a minimum cover and M as a maximum matching
  - Otherwise, select an unmarked $x \in S$ to explore
    - Consider each $y \in N(x)$ such that $xy \notin M$
      - If y is unsaturated, terminate and report an M-augmenting path from U to y
      - Otherwise, $yw \in M$ for some w
        - include y in T (reached from x) and include w in S (reached from y)
    - After exploring all such edges incident to x, mark x and iterate.

Example



Red: A random matching



Theoretical guarantee for Augmenting path algorithm

- Theorem (3.2.2, W) Repeatedly applying the Augmenting Path Algorithm to a bipartite graph produces a matching and a vertex cover of equal size

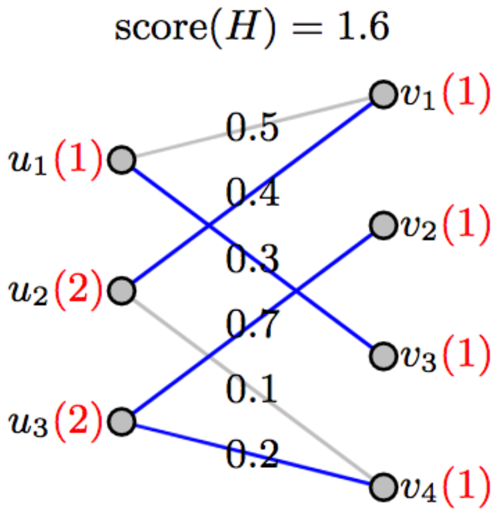
Weighted Bipartite Matching
Hungarian Algorithm

Weighted bipartite matching

- The maximum weighted matching problem is to seek a perfect matching M to maximize the total weight $w(M)$
- Bipartite graph
  - W.l.o.g. Assume the graph is $K_{n,n}$ with $w_{i,j} \geq 0$ for all $i, j \in [n]$
  - Optimization:

$$\max \sum_{i,j} a_{i,j} w_{i,j}$$
$$s.t.\ a_{i,1} + \cdots + a_{i,n} \leq 1 \text{ for any } i$$
$$a_{1,j} + \cdots + a_{n,j} \leq 1 \text{ for any } j$$
$$a_{i,j} \in \{0,1\}$$

  - Integer programming
  - General IP problems are NP-Complete



(Weighted) cover

- A (weighted) cover is a choice of labels $u_1, \dots, u_n$ and $v_1, \dots, v_n$ such that $u_i + v_j \geq w_{i,j}$ for all i, j
  - The cost $c(u, v)$ of a cover (u, v) is $\sum_i u_i + \sum_j v_j$
  - The minimum weighted cover problem is that of finding a cover of minimum cost
- Optimization problem

$$\min \sum_i u_i + \sum_j v_j$$
$$s.t.\ u_i + v_j \geq w_{i,j} \text{ for any } i, j$$
$$u_i, v_j \geq 0 \text{ for any } i, j$$

Duality

(IP)

$$\max \sum_{i,j} a_{i,j} w_{i,j}$$
$$s.t.\ a_{i,1} + \cdots + a_{i,n} \leq 1 \text{ for any } i$$
$$a_{1,j} + \cdots + a_{n,j} \leq 1 \text{ for any } j$$
$$a_{i,j} \in \{0,1\}$$

(Linear programming)

$$\max \sum_{i,j} a_{i,j} w_{i,j}$$
$$s.t.\ a_{i,1} + \cdots + a_{i,n} \leq 1 \text{ for any } i$$
$$a_{1,j} + \cdots + a_{n,j} \leq 1 \text{ for any } j$$
$$a_{i,j} \geq 0$$

(Dual)

$$\min \sum_i u_i + \sum_j v_j$$
$$s.t.\ u_i + v_j \geq w_{i,j} \text{ for any } i,j$$
$$u_i, v_j \geq 0$$

- Weak duality theorem
  - For each feasible solution a and (u, v)

$$\sum_{i,j} a_{i,j} w_{i,j} \leq \sum_i u_i + \sum_j v_j$$

thus $\max \sum_{i,j} a_{i,j} w_{i,j} \leq \min \sum_i u_i + \sum_j v_j$

Duality (cont.)

- Strong duality theorem
  - If one of the two problems has an optimal solution, so does the other one and that the bounds given by the weak duality theorem are tight

$$\max \sum_{i,j} a_{i,j} w_{i,j} = \min \sum_{i} u_i + \sum_{j} v_j$$

- Lemma (3.2.7, W) For a perfect matching M and cover (u, v) in a weighted bipartite graph G, $c(u, v) \geq w(M)$
  $c(u, v) = w(M) \Leftrightarrow$ M consists of edges $x_i y_j$ such that $u_i + v_j = w_{i,j}$
  In this case, M and (u, v) are optimal.

Equality subgraph

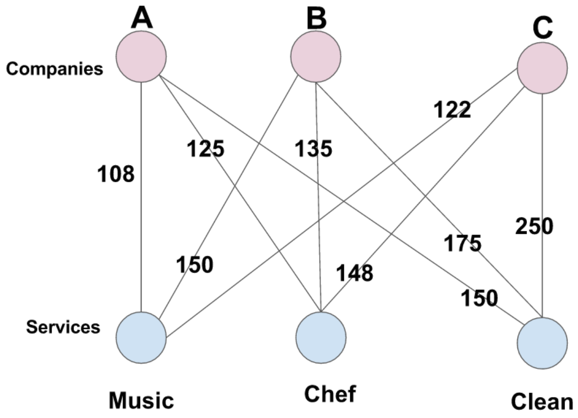
- The equality subgraph $G_{u,v}$ for a cover (u, v) is the spanning subgraph of $K_{n,n}$ having the edges $x_i y_j$ such that $u_i + v_j = w_{i,j}$
  - So if (u, v) is optimal, then M consistes the edges in $G_{u,v}$

Hungarian algorithm

- **Input**: Weighted $K_{n,n} = B(X, Y)$
- **Idea**: Iteratively adjusting the cover (u, v) until the equality subgraph $G_{u,v}$ has a perfect matching
- **Initialization**: Let (u, v) be a cover, such as $u_i = \max_j w_{i,j}$, $v_j = 0$

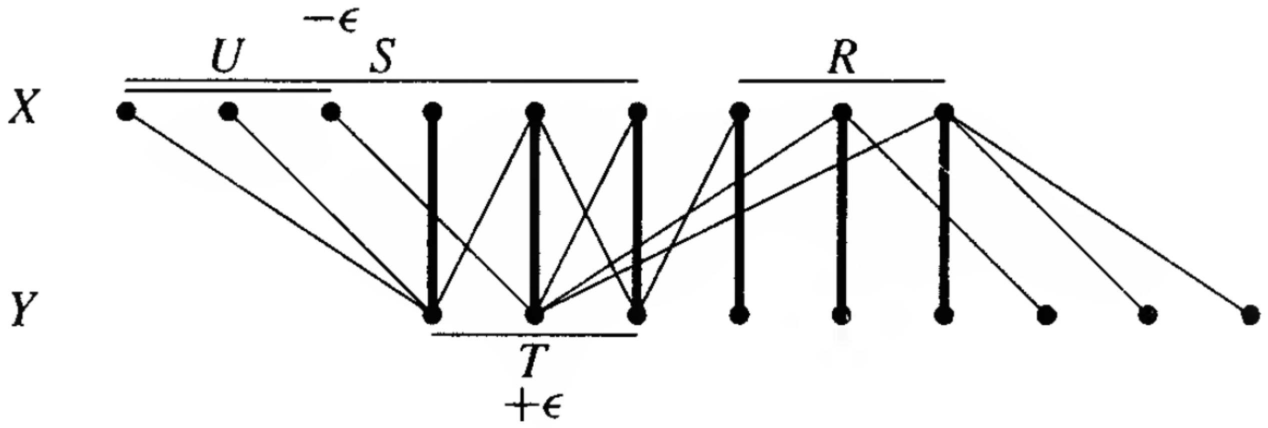
(Dual)

$$\min \sum_i u_i + \sum_j v_j$$
$$s.t. u_i + v_j \geq w_{i,j} \text{ for any } i, j$$
$$u_i, v_j \geq 0$$

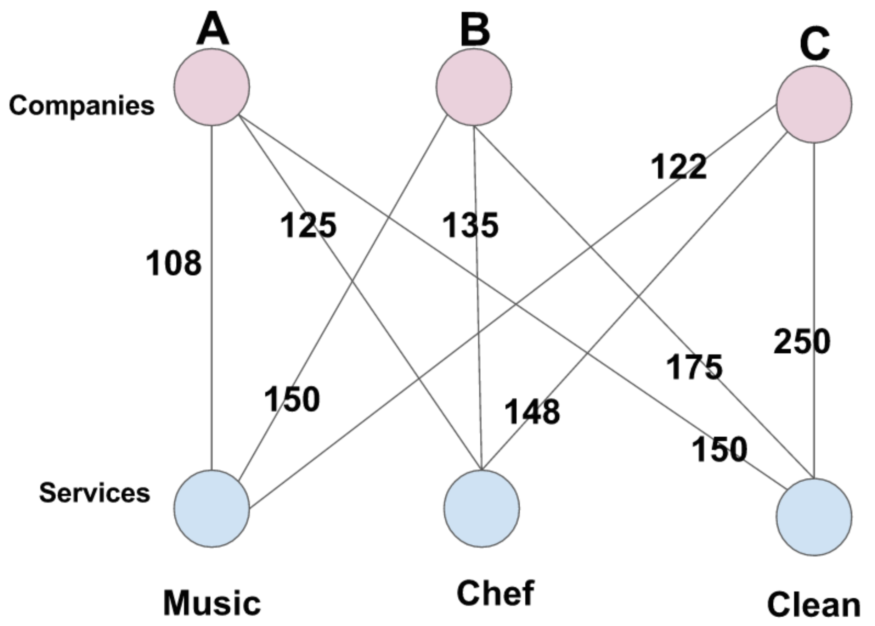


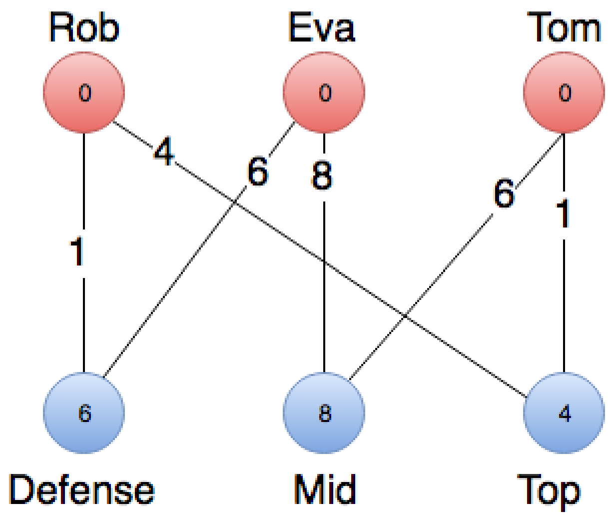
Hungarian algorithm (cont.)

- **Iteration**: Find a maximum matching M in $G_{u,v}$
    - If M is a perfect matching, stop and report M as a maximum weight matching
    - Otherwise, let Q be a vertex cover of size $|M|$ in $G_{u,v}$
        - Let $R = X \cap Q, T = Y \cap Q$
        $$\epsilon = \min\{u_i + v_j - w_{i,j} : x_i \in X - R, y_j \in Y - T\}$$
        - Decrease u_i by ϵ for $x_i \in X - R$ and increase v_j by ϵ for $y_j \in T$
    - Form the new equality subgraph and repeat



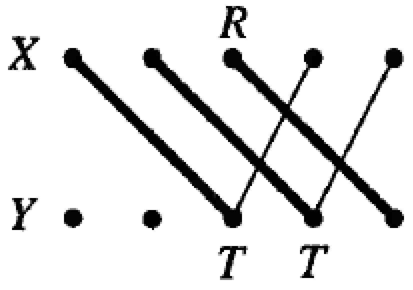
Example


A
B
C
Companies
108
125
135
122
250
175
150
148
150
Services
Music
Chef
Clean

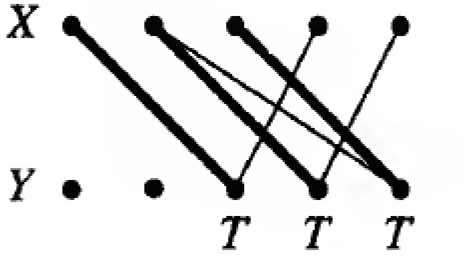

Rob
Eva
Tom
0
0
0
4
6
8
6
1
1
6
8
4
Defense
Mid
Top

Example 2

$$
\begin{pmatrix} 4 & 1 & 6 & 2 & 3 \\ 5 & 0 & 3 & 7 & 6 \\ 2 & 3 & 4 & 5 & 8 \\ 3 & 4 & 6 & 3 & 4 \\ 4 & 6 & 5 & 8 & 6 \end{pmatrix}
\rightarrow
\begin{array}{c} \\ 6 \\ 7 \\ 8 \\ 6 \\ 8 \\ \\ \end{array}
\begin{array}{c} 0 \quad 0 \quad 0 \quad 0 \quad 0 \\ \begin{pmatrix} 2 & 5 & \underline{0} & 4 & 3 \\ 2 & 7 & 4 & \underline{0} & 1 \\ 6 & 5 & 4 & 3 & \underline{0} \\ 3 & 2 & 0 & 3 & 2 \\ 4 & 2 & 3 & 0 & 2 \end{pmatrix} \\ \qquad\quad T \quad T \qquad \end{array}
\begin{array}{c} \\ \\ \\ R \\ \\ \\ \\ \end{array}
$$



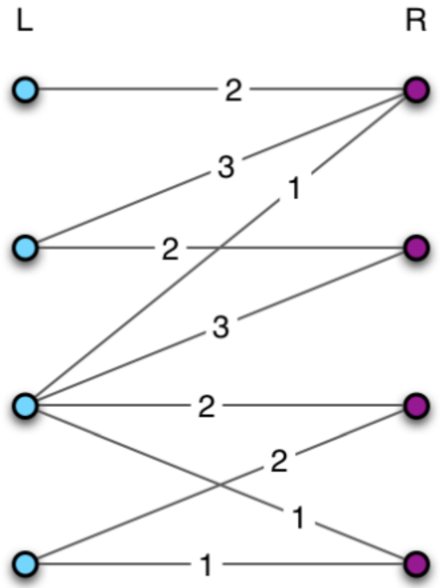
$$
\begin{array}{c} \\ 5 \\ 6 \\ 8 \\ 5 \\ 7 \\ \\ \end{array}
\begin{array}{c} 0 \quad 0 \quad 1 \quad 1 \quad 0 \\ \begin{pmatrix} 1 & 4 & \underline{0} & 4 & 2 \\ 1 & 6 & 4 & \underline{0} & 0 \\ 6 & 5 & 5 & 4 & \underline{0} \\ 2 & 1 & 0 & 3 & 1 \\ 3 & 1 & 3 & 0 & 1 \end{pmatrix} \\ \qquad\quad T \quad T \quad T \end{array}
$$



$$
\rightarrow
\begin{array}{c} \\ 4 \\ 5 \\ 7 \\ 4 \\ 6 \end{array}
\begin{array}{c} 0 \quad 0 \quad 2 \quad 2 \quad 1 \\ \begin{pmatrix} 0 & 3 & \underline{0} & 4 & 2 \\ \underline{0} & 5 & 4 & 0 & 0 \\ 5 & 4 & 5 & 4 & \underline{0} \\ 1 & \underline{0} & 0 & 3 & 1 \\ 2 & 0 & 3 & \underline{0} & 1 \end{pmatrix} \end{array}
$$

Optimal value is the same
But the solution is not unique

Example 3



Theoretical guarantee for Hungarian algorithm

- Theorem (3.2.11, W) The Hungarian Algorithm finds a maximum weight matching and a minimum cost cover

Back to (unweighted) bipartite graph

- The weights are binary $0,1$
- Hungarian algorithm always maintain integer labels in the weighted cover, thus the solution will always be $0,1$
- The vertices receiving label 1 must cover the weight on the edges, thus cover all edges
- So the solution is a minimum vertex cover

Stable Matchings

Stable matching

- A family $(\leq_v)_{v \in V}$ of linear orderings $\leq_v$ on $E(v)$ is a set of preferences for G
- A matching M in G is stable if for any edge $e \in E \setminus M$, there exists an edge $f \in M$ such that e and f have a common vertex v with $e <_v f$
  - Unstable: There exists $xy \in E \setminus M$ but $xy', x'y \in M$ with $xy' <_x xy$ $x'y <_y xy$

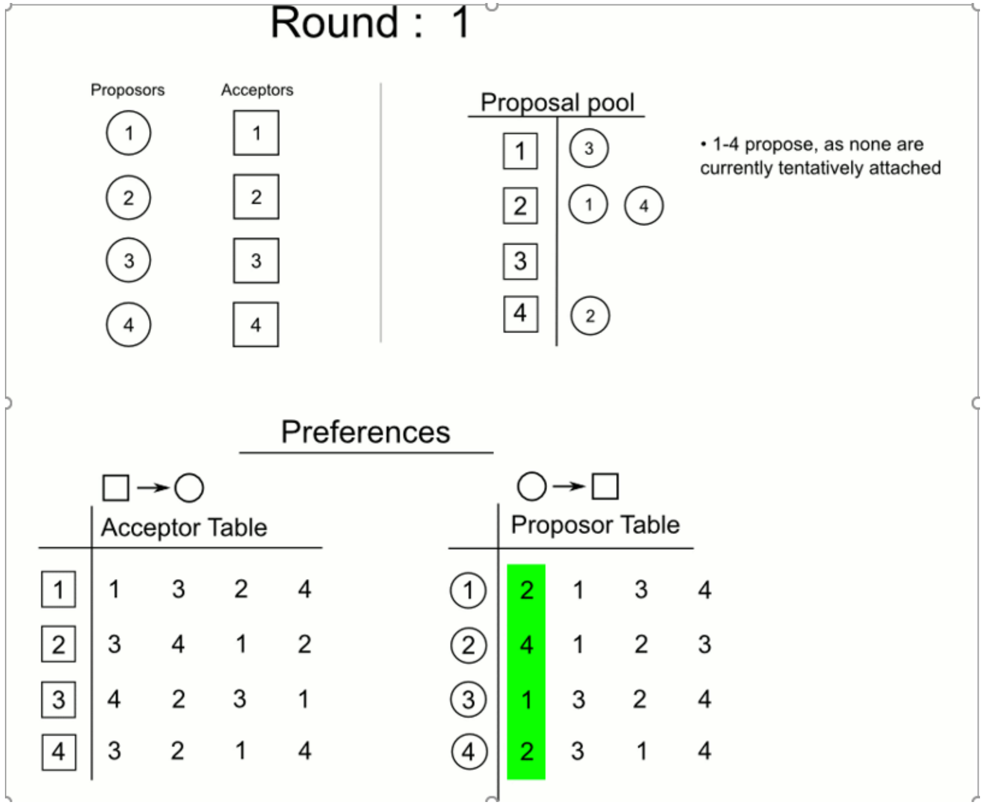
3.2.16. Example. Given men x, y, z, w, women a, b, c, d, and preferences listed below, the matching $\{xa, yb, zd, wc\}$ is a stable matching. ■

Men $\{x, y, z, w\}$	Women $\{a, b, c, d\}$
$x : a > b > c > d$	$a : z > x > y > w$
$y : a > c > b > d$	$b : y > w > x > z$
$z : c > d > a > b$	$c : w > x > y > z$
$w : c > b > a > d$	$d : x > y > z > w$

Gale-Shapley Proposal Algorithm

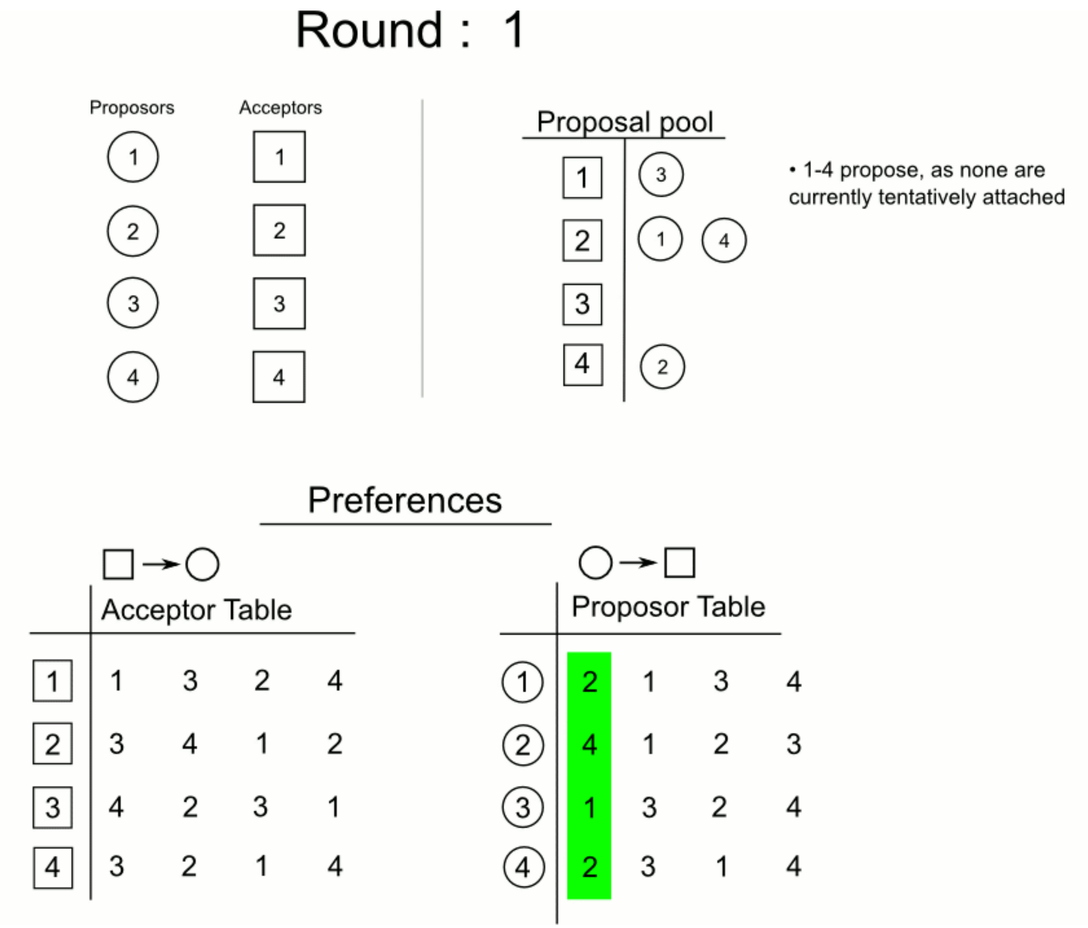
- **Input**: Preference rankings by each of n men and n women
- **Idea**: Produce a stable matching using proposals by maintaining information about who has proposed to whom and who has rejected whom
- **Iteration**: Each man proposes to the highest woman on his preference list who has not previously rejected him
  - If each woman receives exactly one proposal, stop and use the resulting matching
  - Otherwise, every woman receiving more than one proposal rejects all of them except the one that is highest on her preference list
  - Every woman receiving a proposal says “maybe” to the most attractive proposal received

Example



Round : 1
Proposors
Acceptors
1
2
3
4
Proposal pool
1 3
2 1 4
3
4 2
• 1-4 propose, as none are currently tentatively attached
Preferences
Acceptor Table
1 1 3 2 4
2 3 4 1 2
3 4 2 3 1
4 3 2 1 4
Proposor Table
1 2 1 3 4
2 4 1 2 3
3 1 3 2 4
4 2 3 1 4

Example (gif)


Round : 1
Proposors
Acceptors
1 2 3 4
1 2 3 4
Proposal pool
1 | 3
2 | 1 4
3 |
4 | 2
• 1-4 propose, as none are currently tentatively attached
Preferences
Acceptor Table
1 | 1 3 2 4
2 | 3 4 1 2
3 | 4 2 3 1
4 | 3 2 1 4
Proposor Table
1 | 2 1 3 4
2 | 4 1 2 3
3 | 1 3 2 4
4 | 2 3 1 4

Theoretical guarantee for the Proposal Algorithm

- Theorem (3.2.18, W, Gale-Shapley 1962) The Proposal Algorithm produces a stable matching
- Who proposes matters (jobs/candidates)
- When the algorithm runs with women proposing, every woman is as least as happy as when men do the proposing
  - And every man is at least as unhappy

3.2.16. Example. Given men x, y, z, w, women a, b, c, d, and preferences listed below, the matching $\{xa, yb, zd, wc\}$ is a stable matching. ■

Men $\{x, y, z, w\}$	Women $\{a, b, c, d\}$
$x : a > b > c > d$	$a : z > x > y > w$
$y : a > c > b > d$	$b : y > w > x > z$
$z : c > d > a > b$	$c : w > x > y > z$
$w : c > b > a > d$	$d : x > y > z > w$

Matchings in general graphs

Perfect matchings

- K_{2n}, C_{2n}, P_{2n} have perfect matchings
- Corollary (3.1.13, W; 2.1.3, D) Every k-regular ($k > 0$) bipartite graph has a perfect matching
- Theorem(1.58, H) If G is a graph of order $2n$ such that $\delta(G) \geq n$, then G has a perfect matching

Theorem (1.22, H, Dirac) Let G be a graph of order $n \geq 3$. If $\delta(G) \geq n/2$, then G is Hamiltonian

Tutte's Theorem (TONCAS)

- Let $q(G)$ be the number of connected components with odd order
- Theorem (1.59, H; 2.2.1, D; 3.3.3, W)
  Let G be a graph of order $n \geq 2$. G has a perfect matching $\Leftrightarrow q(G - S) \leq |S|$ for all $S \subseteq V$

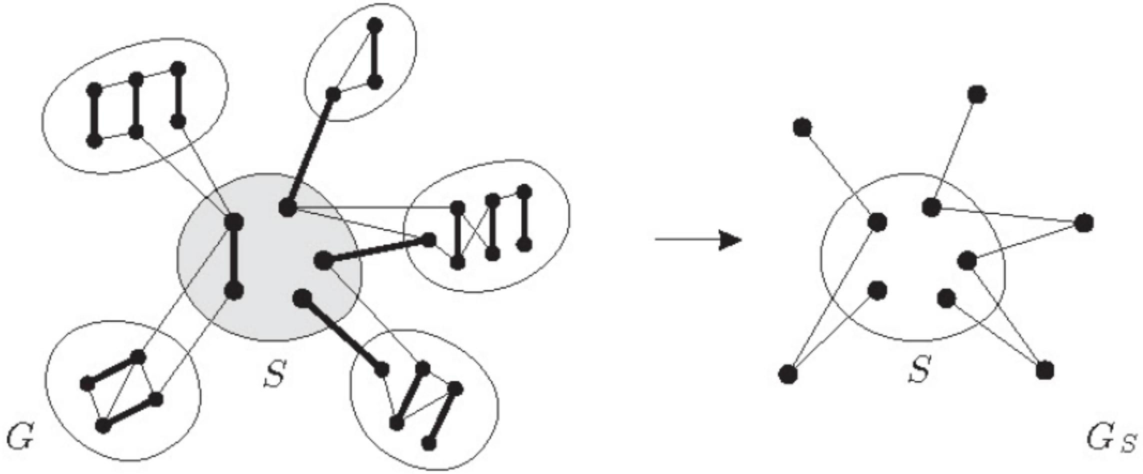


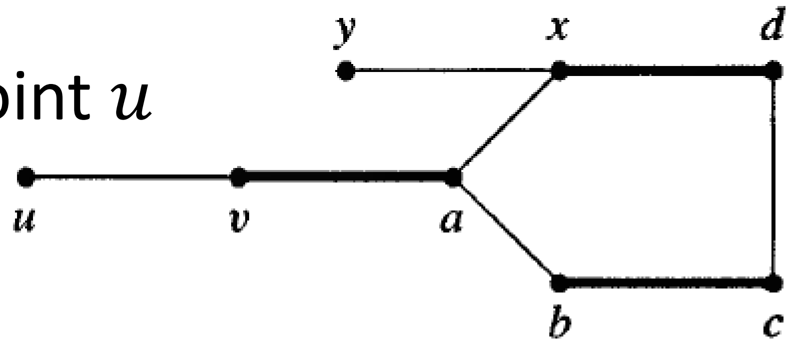
Fig. 2.2.1. Tutte's condition $q(G-S) \leqslant |S|$ for $q = 3$, and the contracted graph G_S from Theorem 2.2.3.

Petersen’s Theorem

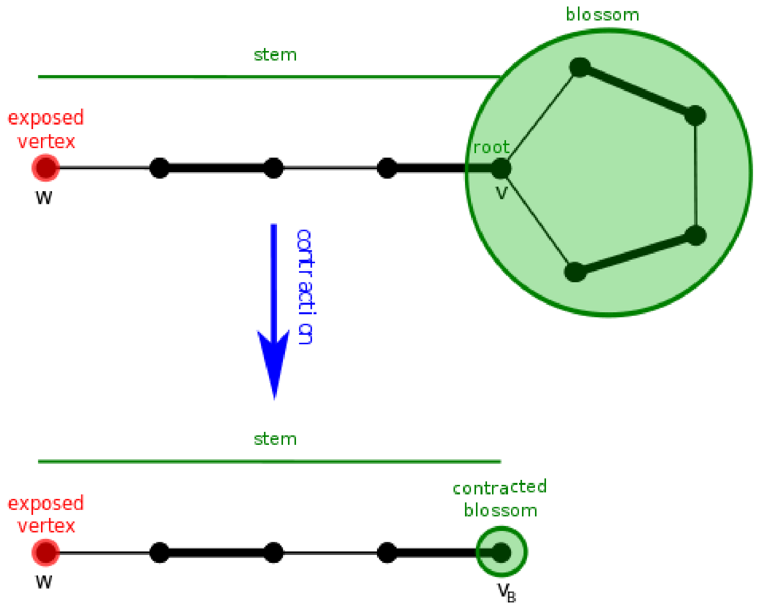
- Theorem (1.60, H; 2.2.2, D;3.3.8, W)
  Every bridgeless, 3-regular graph contains a perfect matching

Find augmenting paths in general graphs

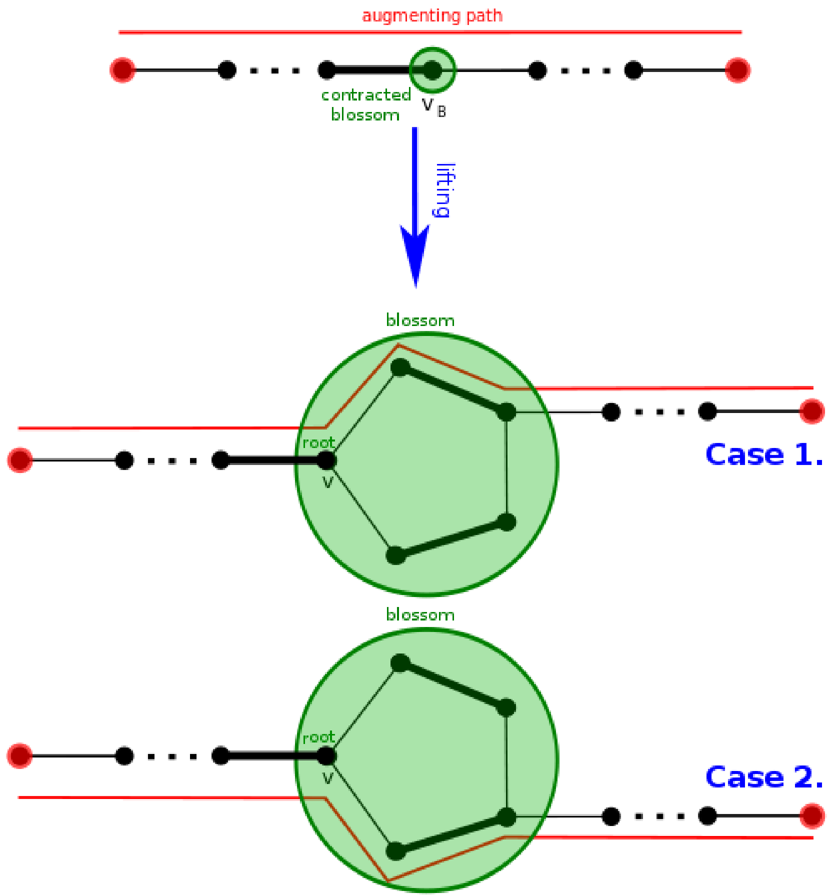
- Different from bipartite graphs
- Example: How to explore from M-unsaturated point u

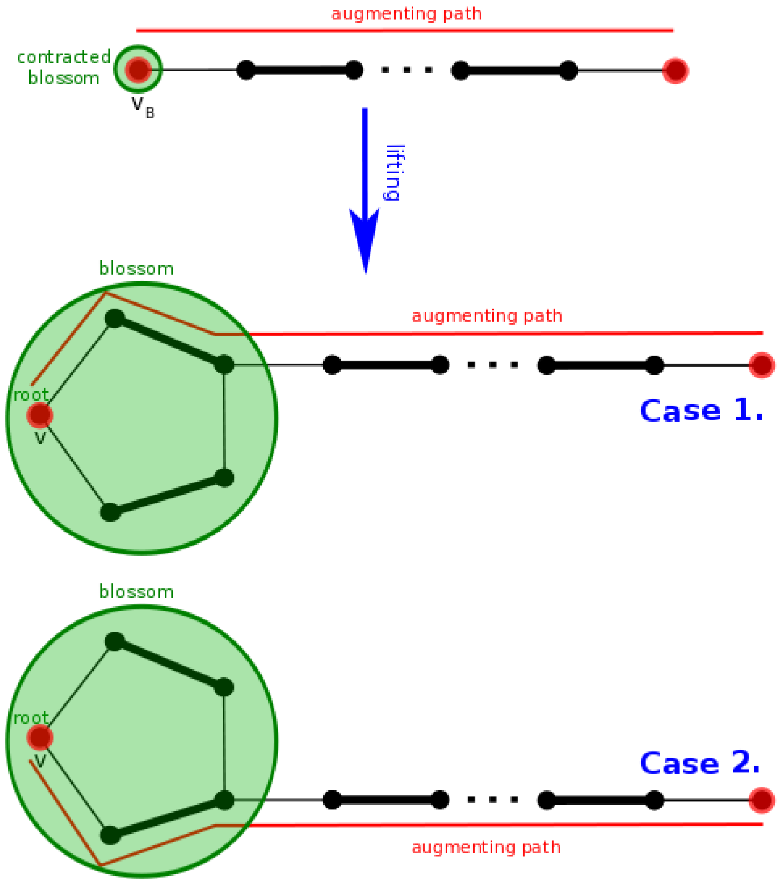


- Flower/stem/blossom



Lifting

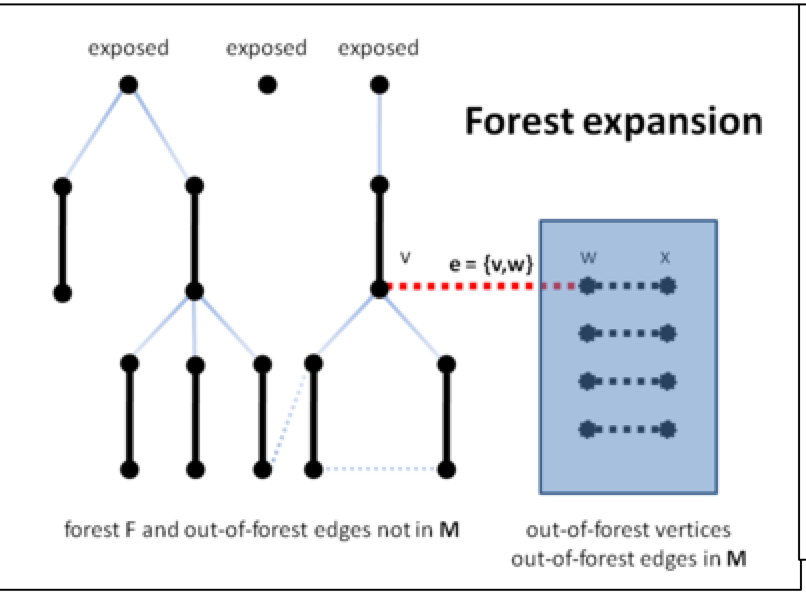

augmenting path
contracted blossom
v_B
lifting
blossom
root
v
Case 1.
blossom
root
v
Case 2.

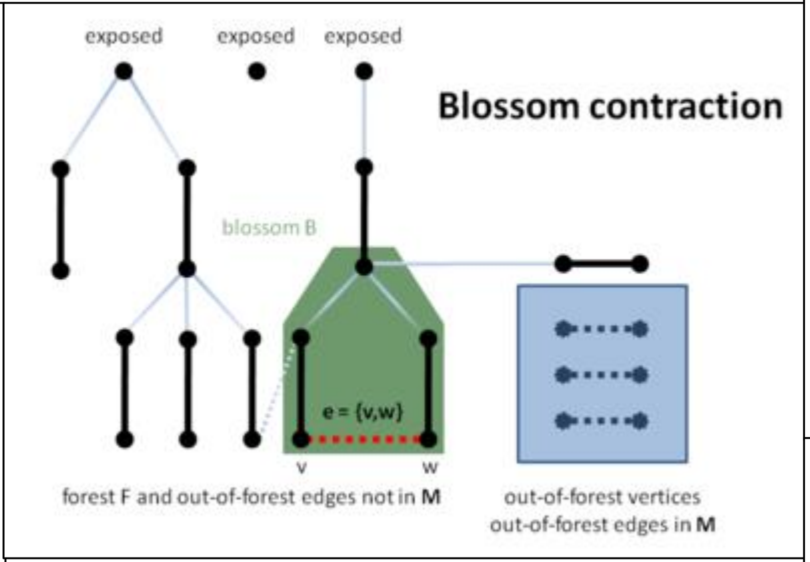

augmenting path
contracted blossom
v_B
lifting
blossom
augmenting path
root
v
Case 1.
blossom
root
v
Case 2.
augmenting path

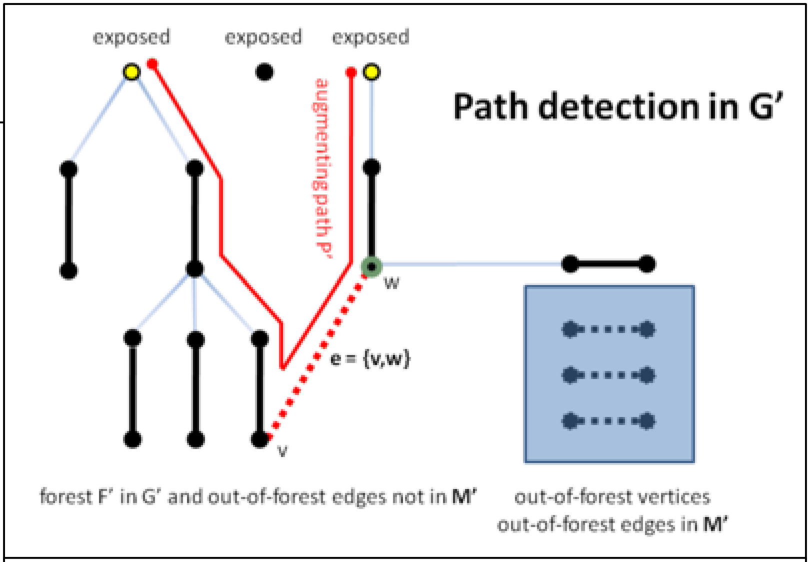
Edmonds' blossom algorithm (3.3.17, W)

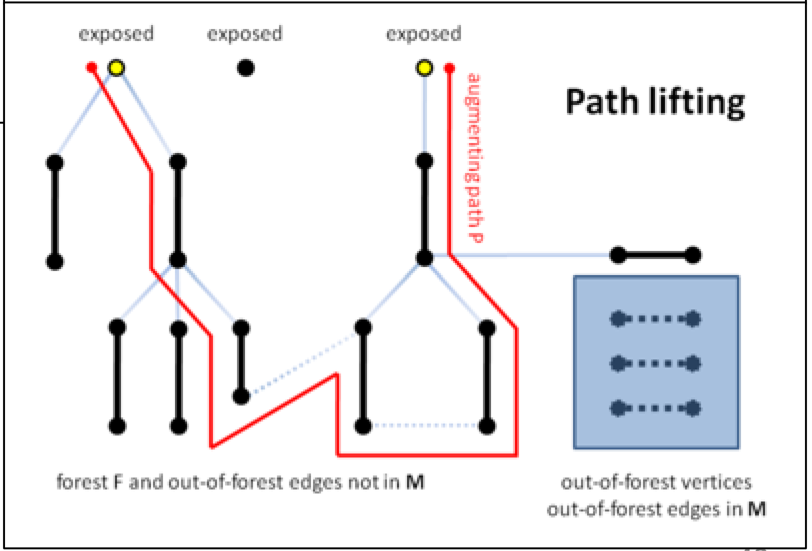
- **Input**: A graph G, a matching M in G, an M-unsaturated vertex u
- **Idea**: Explore M-alternating paths from u, recording for each vertex the vertex from which it was reached, and contracting blossoms when found
  - Maintain sets S and T analogous to those in Augmenting Path Algorithm, with S consisting of u and the vertices reached along saturated edges
  - Reaching an unsaturated vertex yields an augmentation.
- **Initialization**: $S = \{u\}$ and $T = \emptyset$
- **Iteration**: If S has no unmarked vertex, stop; there is no M-augmenting path from u
  - Otherwise, select an unmarked $v \in S$. To explore from v, successively consider each $y \in N(v)$ s.t. $y \notin T$
    - If y is unsaturated by M, then trace back from y (expanding blossoms as needed) to report an M-augmenting u, y-path
    - If $y \in S$, then a blossom has been found. Suspend the exploration of v and contract the blossom, replacing its vertices in S and T by a single new vertex in S. Continue the search from this vertex in the smaller graph.
    - Otherwise, y is matched to some w by M. Include y in T (reached from v), and include w in S (reached from y)
  - After exploring all such neighbors of v, mark v and iterate

Illustration

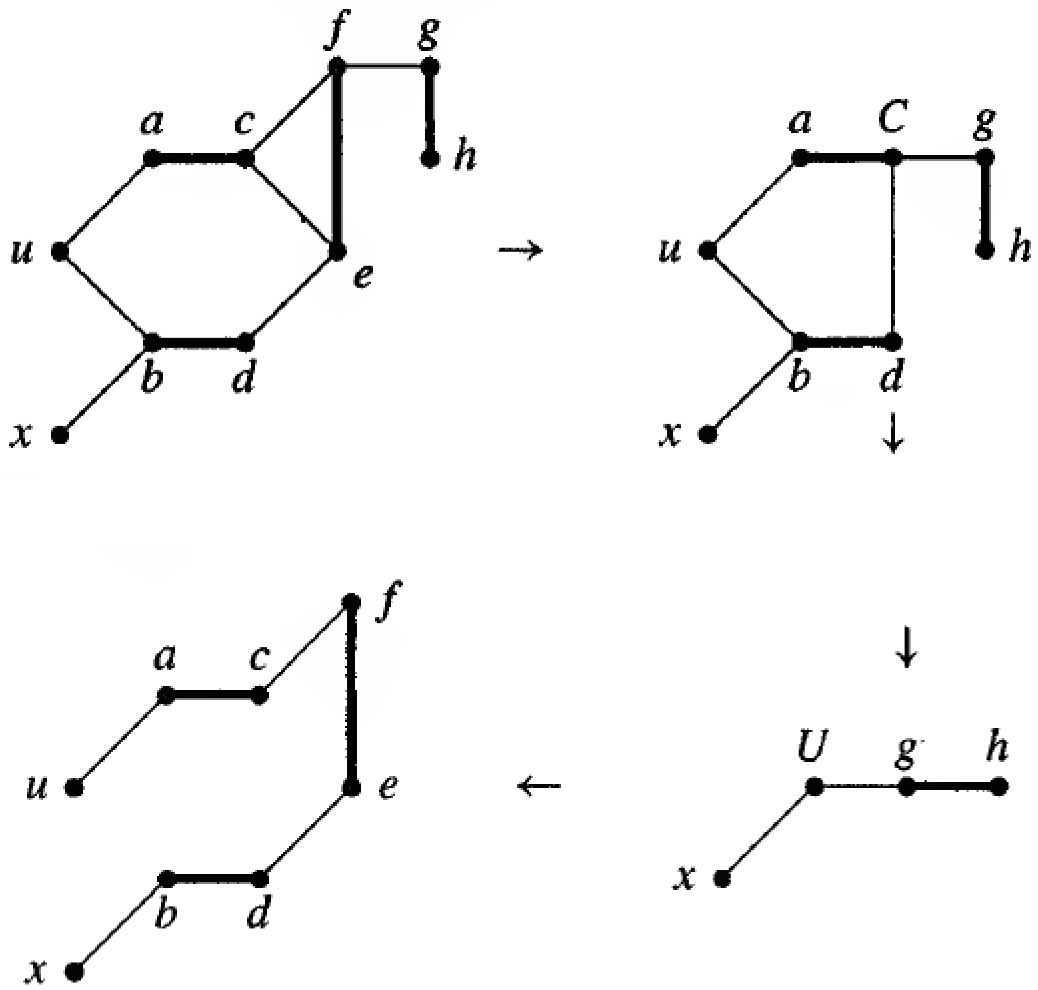




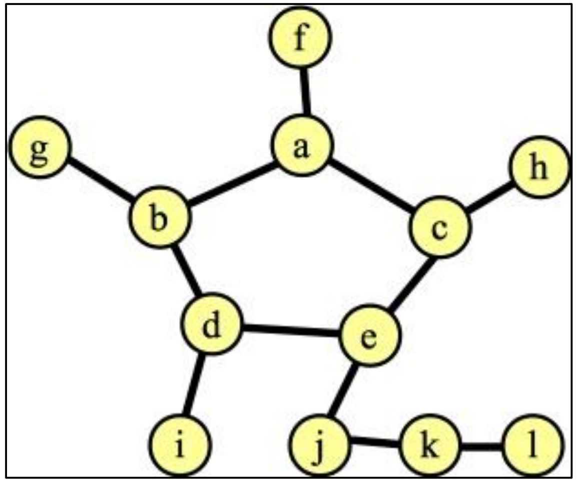


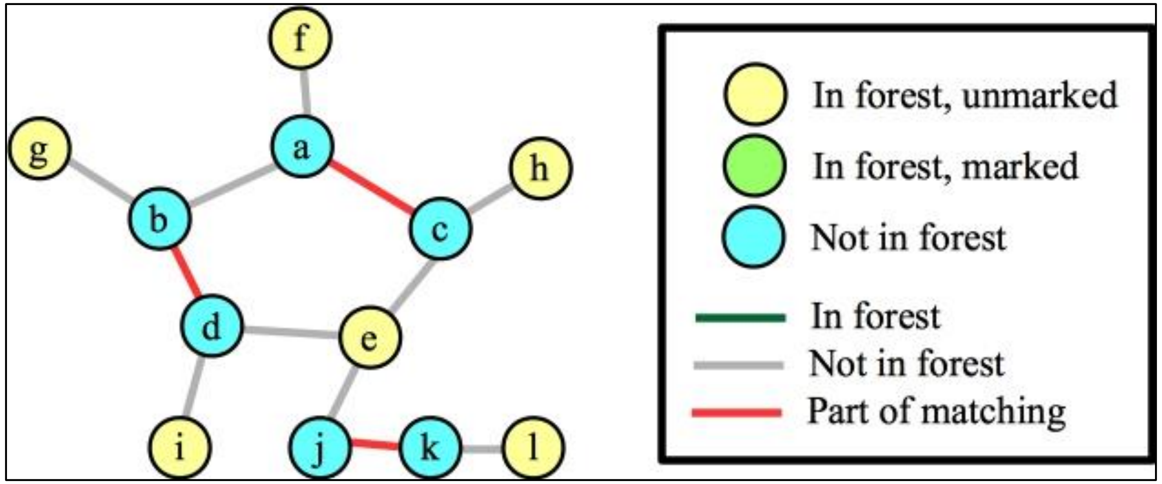


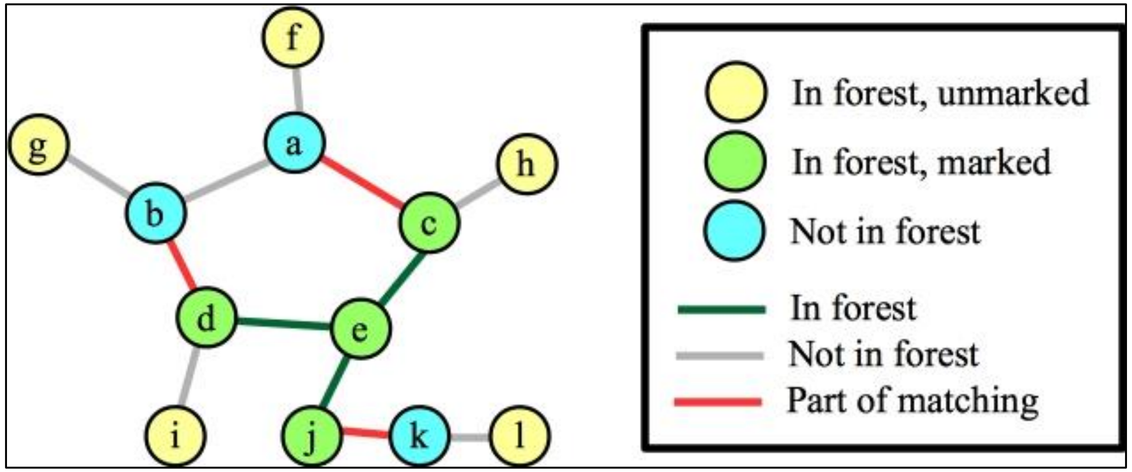
Example



Example 2


f
g
a
h
b
c
d
e
i
j
k
l


f
g
a
h
b
c
d
e
i
j
k
l
In forest, unmarked
In forest, marked
Not in forest
In forest
Not in forest
Part of matching


f
g
a
h
b
c
d
e
i
j
k
l
In forest, unmarked
In forest, marked
Not in forest
In forest
Not in forest
Part of matching

Example 2 (cont.)