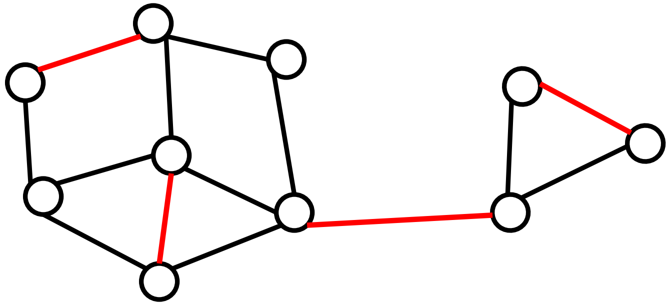
Matchings: Max Cardinality and Min Cost

CSC 473 Advanced Algorithms

Computer Science
UNIVERSITY OF TORONTO

Matchings in graphs

- A matching in a graph $G = (V, E)$ is a subset $M \subseteq E$ of edges so that no two edges in M share an endpoint.



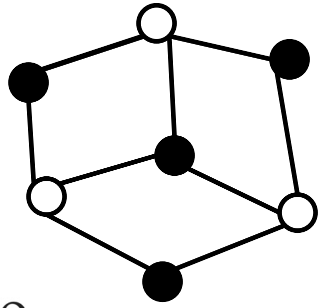
- <u>Maximum cardinality matching</u>: given input graph G, find a matching M of maximum size
  - Perfect matching: size $\frac{|V|}{2}$, i.e., all edges are matched
  - Solvable in polynomial time

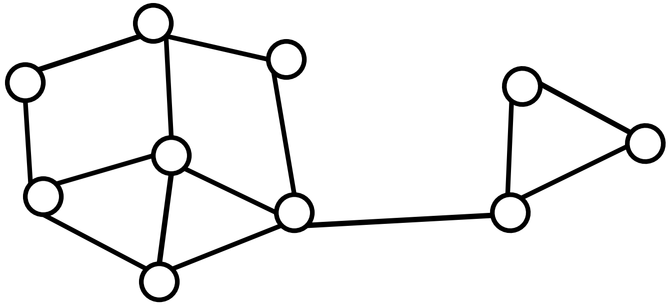
Bipartite Graphs

Algorithm for general graphs is a deep result of Jack Edmonds.

- We will focus on max cardinality matching in *bipartite graphs*.
- <u>Bipartite graph</u>: $G = (V, E)$ so that we can partition V into disjoint sets A and B, and all edges in E have one endpoint in A and on in B
  - We can check if G is bipartite in time $O(n + m)$
  - If it is, we can also find A and B in this time
- *Fact*: a graph is bipartite if and only if it does *not* have an odd cycle
  - *only if* : any path alternates A and B and can only come back to the starting node after an even numbers of hops.

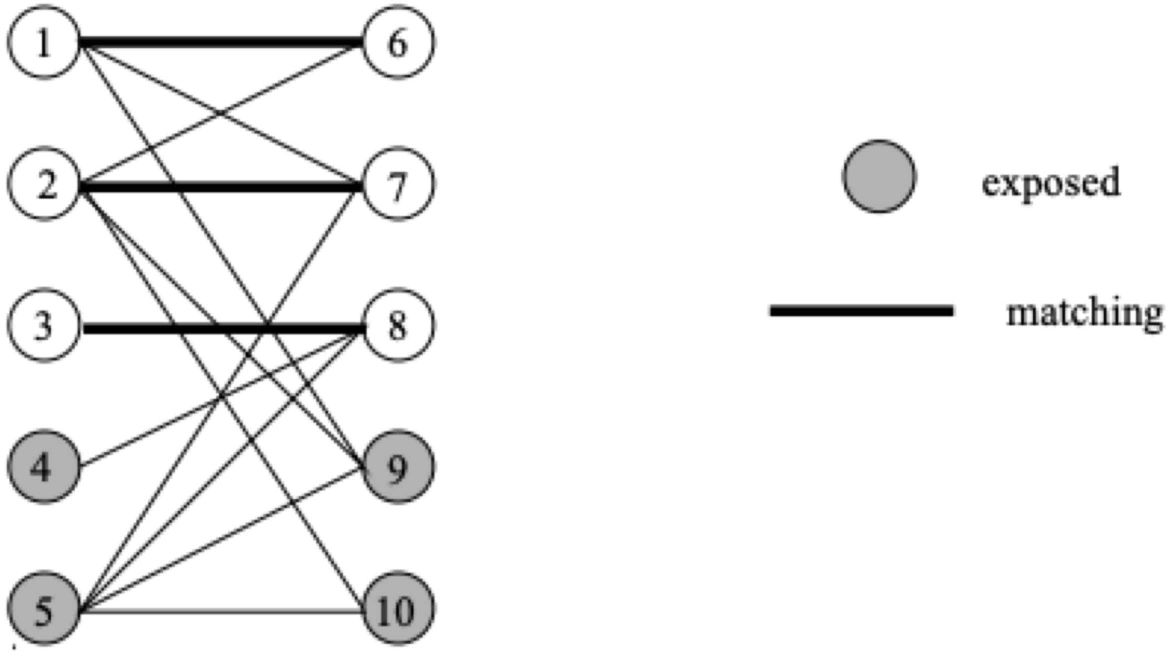
Bipartite





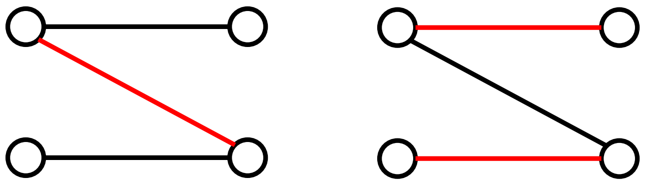
Not bipartite

Bipartite Matching

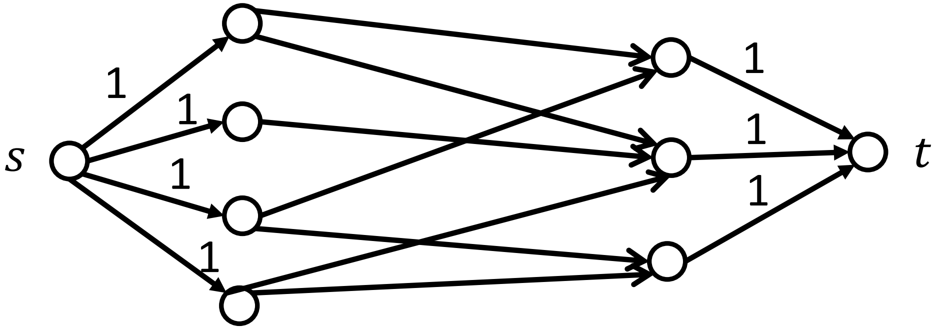


Finding the Maximum Matching

- Greedy (keep adding edges while you can) does not work

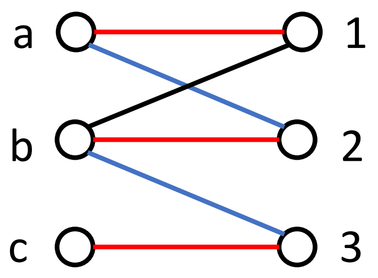


- Do you know a polynomial time algorithm to find the max matching?
  - Compute a max flow
  - We will see a more combinatorial algorithm

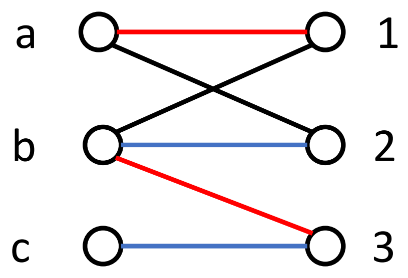


Augmenting Paths

- For a bipartite graph G and a matching M, a path P is alternating if edges in P alternate between being in M and being outside of M.
- An alternating path is augmenting if it starts and ends in unmatched vertices.



1 -> a -> 2 -> b -> 3 -> c
is alternating



2 -> b -> 3 -> c
is augmenting

Edge in M

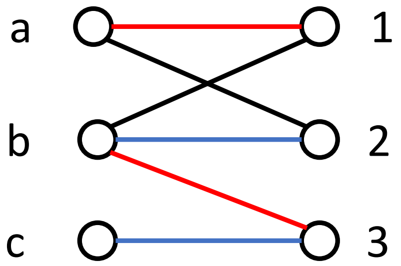
Edge in $P \setminus M$

Augmenting Paths

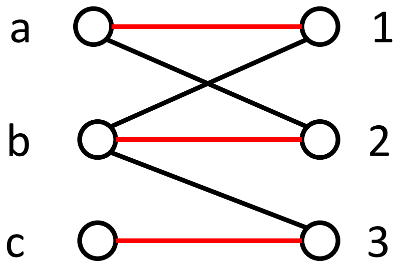
- An alternating path is augmenting if it starts and ends in unmatched vertices.
- Let $P =$ augmenting path. Set $M' = M \triangle P = (M \cup P) \setminus (M \cap P)$.
- M' is a matching and $|M'| = |M| + 1$

First and last vertex in P unmatched, and others just switch which ones they are matched to.

One more non-matching edge in P than matching edges: size of matching increases by 1.



2 -> b -> 3 -> c
is augmenting



Flip which edges are in M along P.

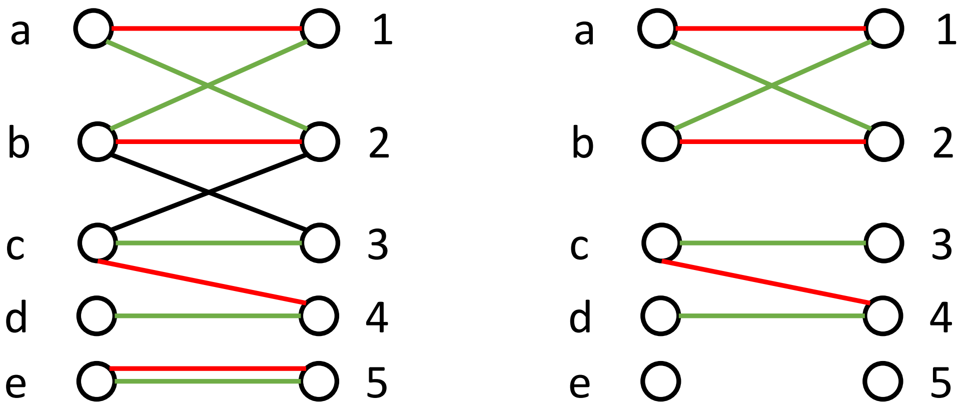
Edge in M

Edge in $P \setminus M$

Characterizing Max Matchings

Theorem. A matching M is of maximum cardinality if and only if there is no augmenting path for it.

- only if: Augmenting path means there is a larger matching
- if: Take a matching M', $|M'| > |M|$, and graph H with edges $M \triangle M'$
  - $M \triangle M' = (M \cup \mathrm{M}') \setminus (M \cap M')$



- H has max degree ≤ 2
- The connected components of H are alternating paths and even cycles
- H has more edges from M', so has a path component starting and ending with edges from M': augmenting for M.

Edge in M

Edge in M'

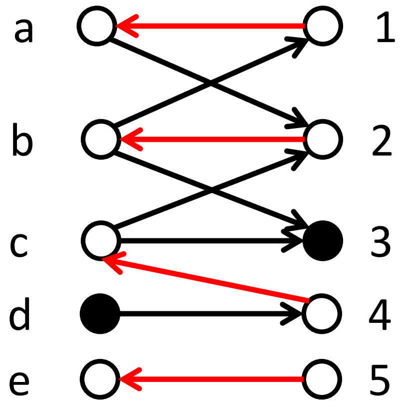
High-Level Algorithm

- $M = \emptyset$
- While $\exists$ an augmenting path P for M
  - Set $M = M \triangle P$

- Correct by Theorem.
- At most $\frac{n}{2}$ iterations, since each iteration adds an edge to M
- How do we search for an augmenting path?
  - Will show a $O(n + m)$ time algorithm, for $O(n^2 + nm)$ total time

Finding Augmenting Paths

- Idea: construct a directed graph $G_M = (V, E_M)$
- alternating paths in G <--> directed paths in G_M



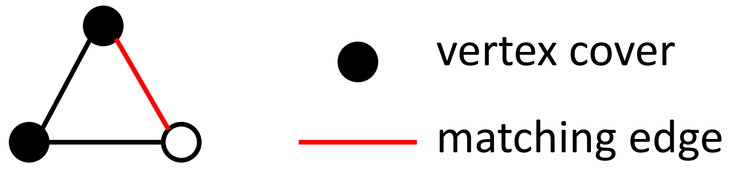
- Direct edges in M from right to left
- Direct edges not in M from left to right
- Any path in G_M must alternate edges in and outside M
- Use BFS to search for a path in G_M from an exposed vertex on the left to an exposed vertex on the right.

König’s Theorem

- Vertex Cover: a set C of vertices of $G = (\mathrm{V}, \mathrm{E})$, so that for any edge $(a, b) \in E$, $a \in C$ or $b \in C$ (or both)

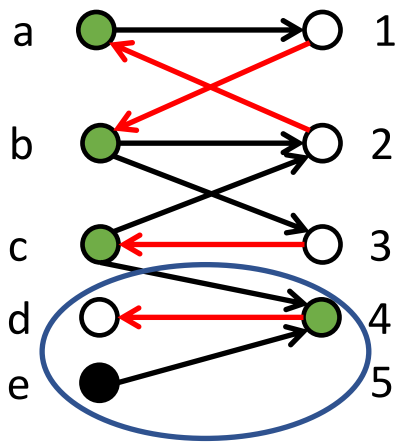
Theorem. In any bipartite graph, the size of the minimum cardinality vertex cover equals the size of the maximum cardinality matching.

- <u>Easy</u>: for any v.c. C and matching M, $|M| \leq |C|$
  - Edges in M are disjoint, so no vertex in C can cover more than one of them
  - True even for non-bipartite graphs.
- <u>Harder</u>: for the max matching M, there exists a v.c. C s.t. $|C| = |M|$
  - This part fails for some non-bipartite graphs.



Proof of Harder Direction

- M = max matching (no augmenting path)
- U = exposed vertices; L = vertices reachable in G_M from $U \cap A$
- $C = (A \setminus L) \cup (B \cap L)$ is a vertex cover of size $|C| = |M|$



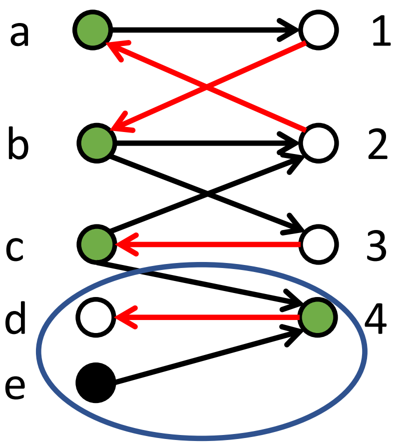
- No edges (a, b) with $a \in A \cap L$ and $b \in B \setminus L$
  - $(a, b) \notin M$: if a is reachable from U, then so is b
  - $(a, b) \in M$: $a \notin U$, and only incoming edge is $a \leftarrow b$, so a is reachable from U only if b is

$L = \{d, e, 4\}$
$C = \{a, b, c, 4\}$

● exposed
● in vertex cover

Proof, continued

- M = max matching (no augmenting path)
- U = exposed vertices; L = vertices reachable in G_M from $U \cap A$
- $C = (A \setminus L) \cup (B \cap L)$ is a vertex cover of size $|C| = |M|$



$L = \{d, e, 4\}$
$C = \{a, b, c, 4\}$

- Every vertex in C touches exactly one edge in M
  - $A \setminus L \subseteq A \setminus U$ so all vertices in $A \setminus L$ are matched
  - All vertices in $B \cap L$ are matched, otherwise there is an augmenting path.
  - If $(a, b) \in M$, and $b \in B \cap L$, then $a \notin A \setminus L$ (if b is reachable from U, so is a).

● exposed
● in vertex cover

Min-Cost Perfect Matching

- Input: a bipartite graph $G = (A \cup B, E)$ which has a perfect matching (matching of size $\frac{n}{2}$), and costs $c \in \mathbb{R}^E$
- Output: a perfect matching M that minimizes $\sum_{e \in M} c_e$.

- We can assume that G is the complete bipartite graph, i.e., there is an edge (a, b) for each $a \in A$, $b \in B$.
  - Set $c(e) = \infty$ for $e \notin E$

LP Relaxation

- Convert into an IP, and relax to an LP

value of the LP
$\leq$ value of the IP
$=$ min cost of a perfect matching

$$\min \sum_{a \in A, b \in B} c_{a,b} x_{a,b}$$

s.t.

$$\sum_{b \in B} x_{a,b} = 1 \quad \forall a \in A$$

$$\sum_{a \in A} x_{a,b} = 1 \quad \forall b \in B$$

$$x_{a,b} \in \{0,1\} \quad \forall a \in A, b \in B$$

Each vertex covered by exactly one matching edge

$x_{a,b} = 1 \Leftrightarrow (a,b) \in M$

$$\min \sum_{a \in A, b \in B} c_{a,b} x_{a,b}$$

s.t.

$$\sum_{b \in B} x_{a,b} = 1 \quad \forall a \in A$$

$$\sum_{a \in A} x_{a,b} = 1 \quad \forall b \in B$$

$$x_{a,b} \geq 0 \quad \forall a \in A, b \in B$$

$x_{a,b} \leq 1$ implied by the other constraints.

The LP is integral

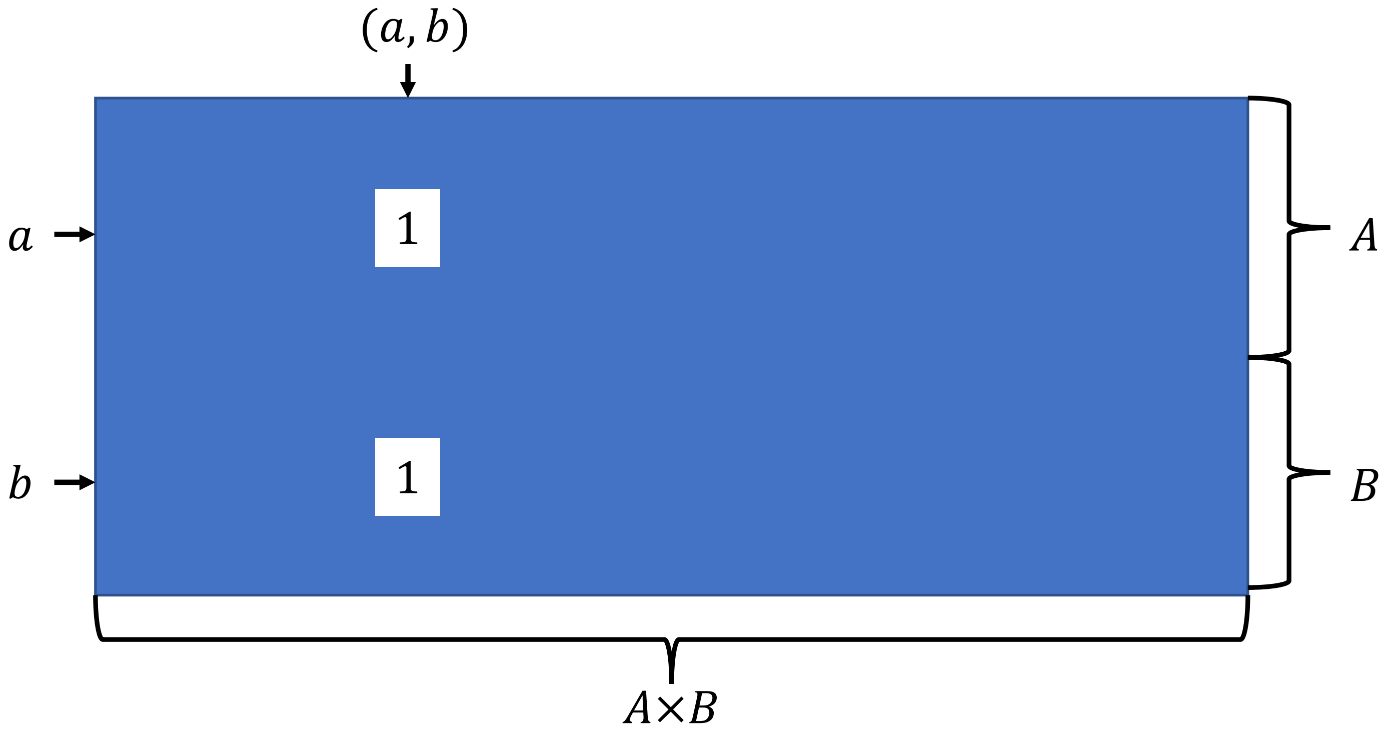
The feasible region of the LP is a polytope whose vertices are indicator vectors of perfect matchings.

Theorem. The value of the LP relaxation is equal to the minimum cost of a perfect matching. Moreover, a min cost matching is computable in time $O(n^3)$.

- Identify LP solutions x with coordinates in $\{0,1\}$ with matchings
  - $x_{a,b} = 1 \Leftrightarrow (a, b) \in M$
- Theorem says that for any cost vector c, there is a $\{0,1\}$-solution x (equivalently a matching M) which is optimal for the LP.

Matrix form

- Can write LP as $\min\{c^\top x\colon Hx = 1, x \geq 0\}$ for $n \times \binom{n}{2}^2$ matrix H:



The Dual

- The primal and dual LPs:

$$\min \sum_{a\in A, b\in B} c_{a,b} x_{a,b}$$

s.t.

$$\sum_{b\in B} x_{a,b} = 1 \quad \forall a \in A$$

$$\sum_{a\in A} x_{a,b} = 1 \quad \forall b \in B$$

$$x_{a,b} \geq 0 \;\; \forall a \in A, b \in B$$

$$\max \sum_{u\in A\cup B} y_u$$

s.t. $\quad y_a + y_b \leq c_{a,b} \quad \forall a \in A, b \in B$

- Complementary Slackness: feasible solutions x and y are optimal iff
  - $x_{a,b} > 0 \Rightarrow y_a + y_b = c_{a,b}$
- Goal: compute feasible y and a p.m. M, s.t. $M \subseteq E_y = \{(a,b): y_a + y_b = c_{a,b}\}$

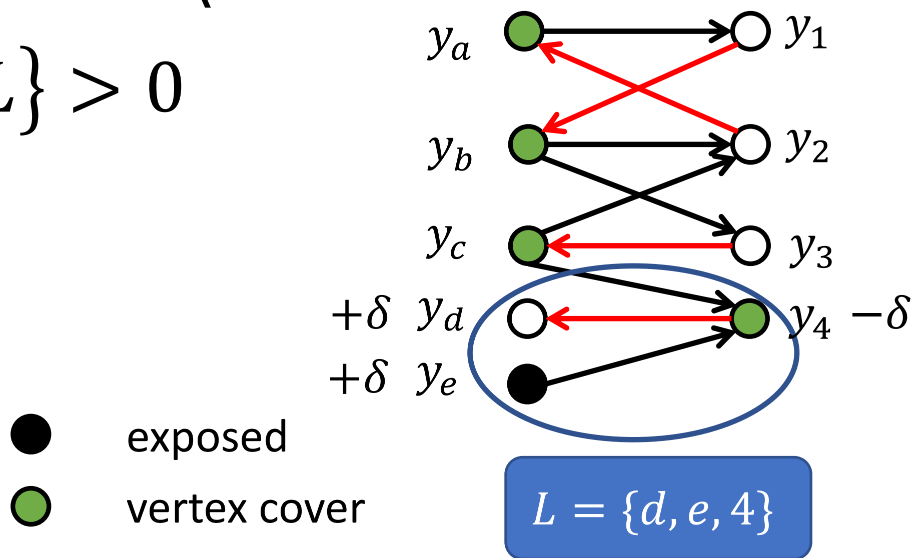
High-Level Algorithm

- Start with $y = 0, M = \emptyset$
- While M is not perfect
  - If there is an augmenting path P in $G_y = (A \cup B, E_y)$
    - $M = M \triangle P$
  - Else, modify y while maintaining $M \subseteq E_y$

- Will make sure that after each $O(n)$ modifications to y, an augmenting path exists (unless M is perfect).
- On termination, by compl. slackness, M is a min cost perfect matching.
  - I.e., the LP solution x corresponding to M and y satisfy CS.

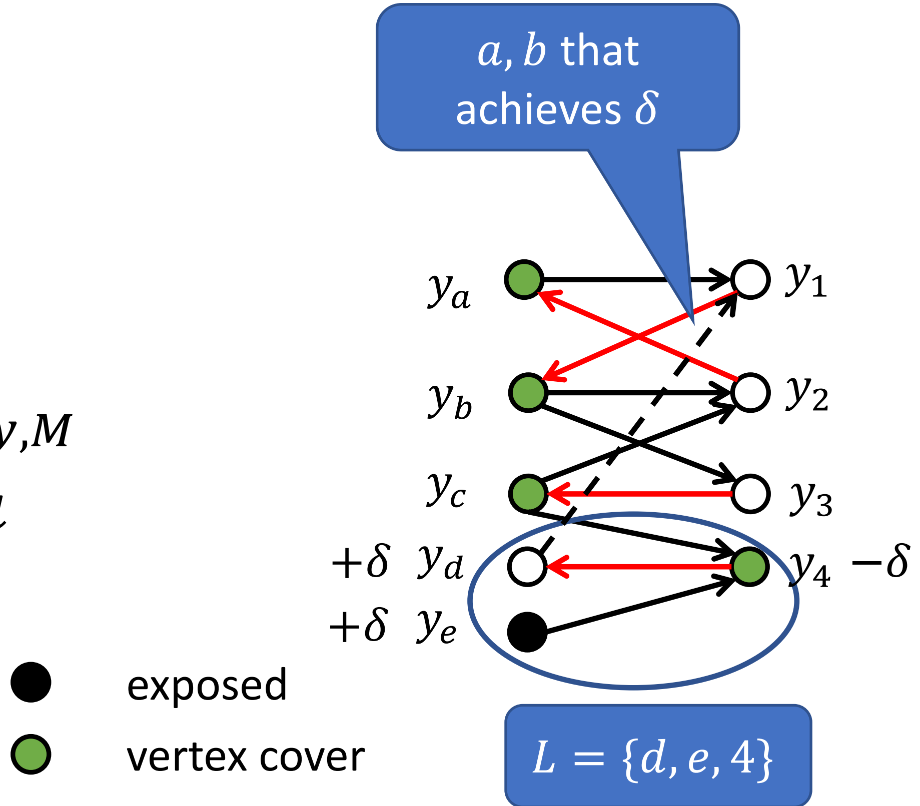
Modifying y

- $G_{y,M}$ = orientation of G_y associated with M
  - i.e. directed graph with edges in M directed to the left, and the others right
- U = exposed vertices; L = vertices reachable in $G_{y,M}$ from $U \cap A$
- Assume no augmenting path, so $L \cap U \cap B = \emptyset$
- König's Thm: No edges of G_y between $A \cap L$ and $B \setminus L$
- $\delta = \min\{c_{a,b} - y_a - y_b : a \in A \cap L, b \in B \setminus L\} > 0$
- Modification:
  - $y_a \leftarrow y_a + \delta \;\; \forall a \in A \cap L$
  - $y_b \leftarrow y_b - \delta \;\; \forall b \in B \cap L$



Modifying y

- $\delta = \min\{c_{a,b} - y_a - y_b : a \in A \cap L, b \in B \setminus L\} > 0$
- Modification:
  - $y_a \leftarrow y_a + \delta \;\; \forall a \in A \cap L$
  - $y_b \leftarrow y_b - \delta \;\; \forall b \in B \cap L$
- y is still feasible (by definition of δ)
- $M \subseteq E_y$ after modification
  - if $(a, b) \in M$ and $b \in B \cap L$, then $a \in A \cap L$
- All reachable vertices remain reachable in new $G_{y,M}$
- For a, b achieving δ, b becomes reachable from a
  - $b \in L$



Running time

- After each modification to y, a new vertex in B enters L
- After $\leq \frac{n}{2}$ modifications, some exposed vertex in B enters L
  - I.e., there is a an augmenting path
- So after each $\leq \frac{n}{2}$ modifications to y, M grows by 1 edge
- Total number of iterations is $O(n^2)$, each taking $O(n^2)$ time
  - Running time $O(n^4)$
- Better data structures improve this to $O(n^3)$