Combinatorial Mathematics

Mong-Jen Kao (高孟駿)

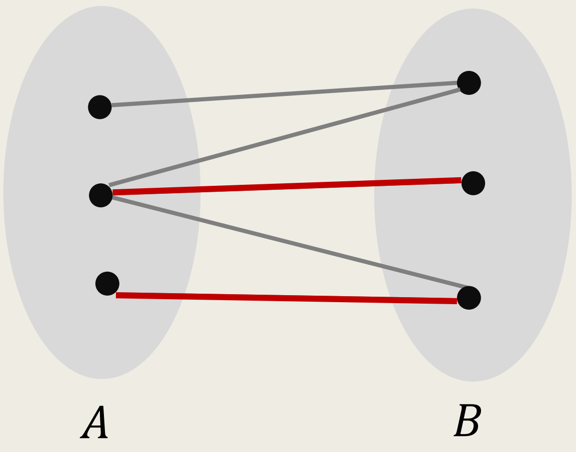
Monday 18:30 – 21:20

Outline

- Hall’s Matching Theorem
- König-Egeváry Theorem
- The Maximum Matching Problem
  - A Generic Algorithm and the Berge’s Theorem
  - The Augmenting Path Problem in Bipartite Graphs
    - A simple DFS-based recursive algorithm

Matching in Bipartite Graphs

- Let $G = (V, E)$ be a bipartite graph with partite sets A and B.
- An edge subset $M \subseteq E$ is called a matching for G,
  if each vertex in V is incident to at most one edge in M.
  - i.e., the endpoints of the edges in M are disjoint.



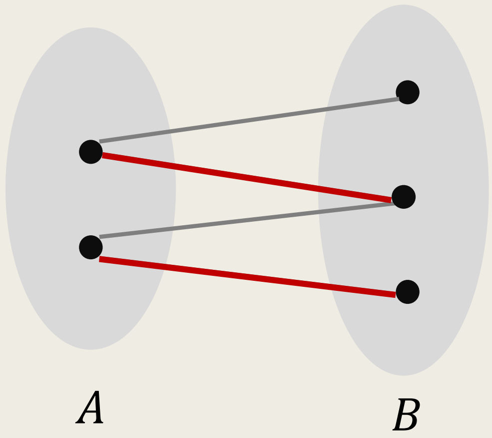
The same definition applies to general graphs, too.

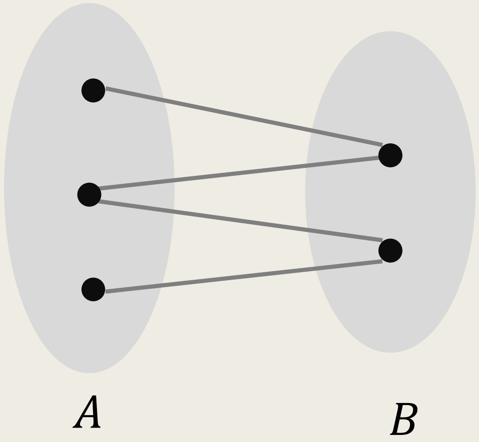
Matching in Bipartite Graphs

- Let $G = (V, E)$ be a bipartite graph with partite sets A and B.
- Let M be a matching for G.
  - For any $u, v \in V$,
    we say that u is matched to v by M (and vice versa),
    if $(u, v) \in M$.
  - For any $U \subseteq A$, we say that M matches U, or,
    M is a matching from U to B, or, M is a matching for U,
    if M matches every vertex in U to some vertex in B.

Matching in Bipartite Graphs

- Let $G = (V, E)$ be a bipartite graph with partite sets A and B.
- Let M be a matching for G.
  - For any $U \subseteq A$, we say that M is a matching for U, if M matches every vertex in U to some vertex in B.





There is no enough candidates to be matched to for A.

Hall’s Matching Condition

The necessary and sufficient condition for a matching in bipartite graphs to exist.

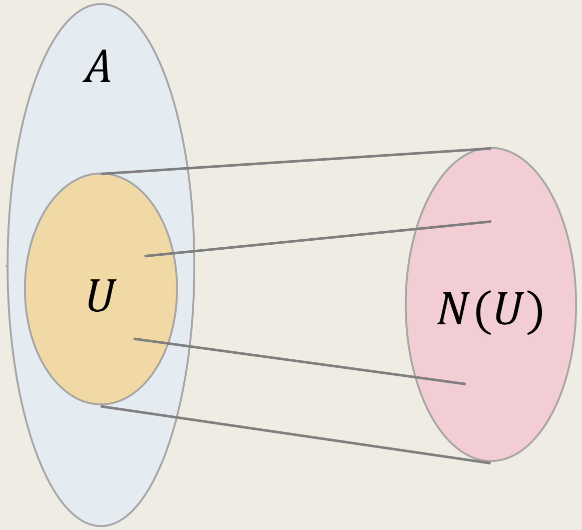
<u>Theorem 5.1 (Hall's Theorem).</u>

Let $G = (V, E)$ be a bipartite graph with partite sets A and B.

There exists a matching M for A

if and only if

$$|N(U)| \geq |U| \quad \text{for all } U \subseteq A. \qquad (*)$$



i.e., there is always a sufficient number of candidates to be matched to.

$|N(U)| \geq |U|$, for any $U \subseteq A$.

Theorem 5.1 (Hall's Theorem).

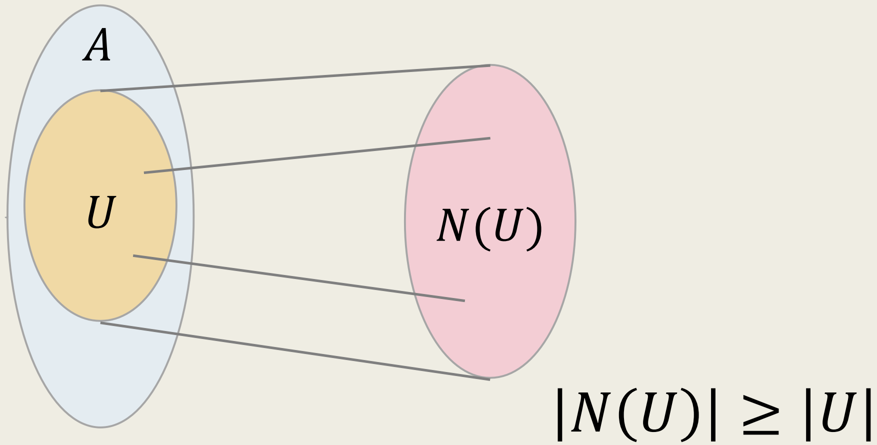
Let $G = (V, E)$ be a bipartite graph with partite sets A and B.

There exists a matching M for A

if and only if

$$|N(U)| \geq |U| \quad \text{for all } U \subseteq A. \qquad (*)$$

- Proof.
  - The direction ($\Longrightarrow$) is clear.
    - M matches each vertex in U to a distinct vertex in B.
    - Hence, $|N(U)| \geq |U|$.



- <u>Proof. (continue)</u>
  - We prove the direction ($\Longleftarrow$) by induction on the size of $|A|$, which we denote by m.
  - The case $m = 1$ holds trivially.
  - Assume that the statement ($\Longleftarrow$) holds for any A with $|A| < m$.

- <u>Proof. (continue)</u>
  - Assume that the statement ($\Longleftarrow$) holds when the number of vertices in the left partite set is $< m$.
  - To prove for $|A| = m$, we distinguish the following two cases.
    1. For any $U \subset A$, we always have $|N(U)| > |U|$.
    2. For some $U \subset A$, $|N(U)| = |U|$.

We *<u>always</u>* have *more candidates* than we need.

The number of candidates for some subset is tight.

- We distinguish following two cases.

We always have more candidates than we need.

1. For any $U \subset A$, we always have $|N(U)| > |U|$.
   - Pick an arbitrary $u \in A$ and any $v \in N(u)$. Match u to v and remove v from the graph.
   - Then, it follows that, for any $U \subseteq A - \{u\}$, we still have $|N(U)| \geq |U|$.
     - By the induction hypothesis, there exists a matching from $A - \{u\}$ to $B - \{v\}$.
     - Hence, we obtain a matching for A.

At most one vertex is removed from $N(U)$.

- We distinguish following two cases.

The number of candidates for some subset is tight.

2. For some $U \subset A$, $|N(U)| = |U|$.

   - By the induction hypothesis, there exists a matching M_1 from U to $N(U)$.

   Remove $N(U)$ from the graph.

   - Then, we claim that, for any $U' \subseteq A - U$, we still have $|N(U')| \geq |U'|$.

- We distinguish following two cases.

The number of candidates for some subset is tight.

2. For some $U \subset A$, $|N(U)| = |U|$.

    - Remove $N(U)$ from the graph.

    - Then, we claim that,
      for any $U' \subseteq A - U$, we always have $|N(U')| \geq |U'|$.

        - If not, then before $N(U)$ is removed, we have

$$|N(U' \cup U)| \leq |N(U')| + |N(U)| < |U'| + |U|,$$

which is a contradiction.

- We distinguish following two cases.

The number of candidates for some subset is tight.

2. For some $U \subset A$, $|N(U)| = |U|$.

   - By the induction hypothesis, there exists a matching M_1 from U to $N(U)$.

     Remove $N(U)$ from the graph.

   - Then, we claim that, for any $U' \subseteq A - U$, we always have $|N(U')| \geq |U'|$.

   - By induction hypothesis, there exists a matching M_2 for $A - U$.

   - Together, we obtain a matching for A.

Application -

System of Distinct Representatives

Distinct Representative of Sets in a Family

- Let $F = \{S_1, S_2, \dots, S_m\}$ be a set family.

- The elements $x_1, x_2, \dots, x_m$ is called a set of *distinct representatives* for F, if the following two conditions hold.
  - $x_i \in S_i$ for all $1 \leq i \leq m$.
  - The elements $x_1, x_2, \dots, x_m$ are distinct, i.e., $x_i \neq x_j$ for all $i \neq j$.

Corollary.

The set family $S_1, S_2, \ldots, S_m$ has a set of distinct representatives ***if and only if***

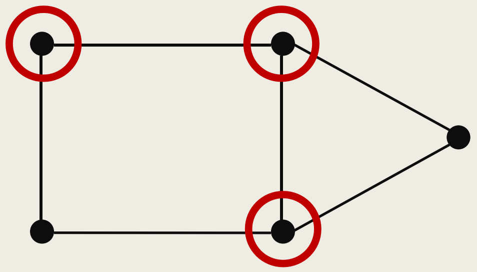
$$\left|\bigcup_{i \in I} S_i\right| \geq |I| \quad \text{for all } I \subseteq \{1,2,\ldots,m\}.$$

- Construct a bipartite graph for the set family, and this corollary follows directly from the Hall's theorem.

Matching v.s. Vertex Cover

Weak-duality between matching and vertex cover.

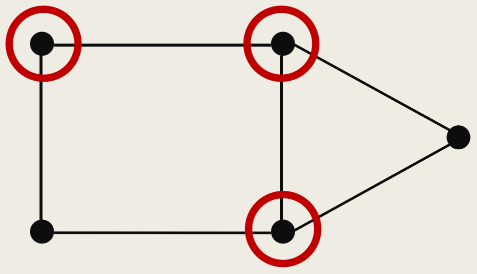
Vertex Cover of a Graph



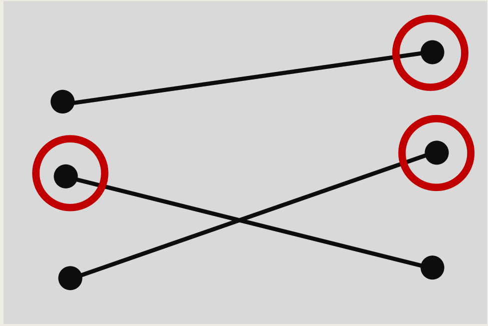
- Let $G = (V, E)$ be a graph.

- A ***vertex cover*** of G is a subset $U \subseteq V$ of vertices such that, every edge $e \in E$ has at least one endpoint in U.
  - Intuitively, we use the vertices in U to cover the edges in E.

Matching v.s. Vertex Cover



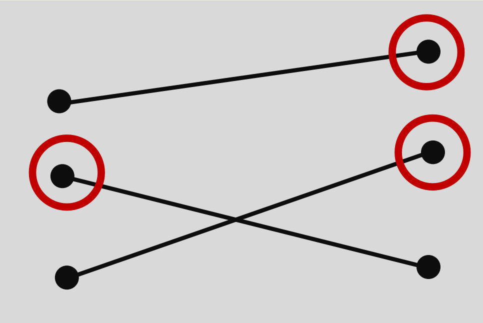
- Let $G = (V, E)$ be a graph,
  - $M \subseteq E$ be a matching, and
  - $C \subseteq V$ be a vertex cover for G.



The matching M

- It follows that
$$|\mathrm{M}| \leq |C|.$$
  - The endpoints of the edges in M are distinct.
  - It takes <u>at least one vertex</u> to cover <u>each edge in M</u>, i.e., at least one endpoint of each edge has to be selected in C.

Matching v.s. Vertex Cover



The matching M

- Let $G = (V, E)$ be a graph,
  $M \subseteq E$ be a matching, and $C \subseteq V$ be a vertex cover for G.

- Then, it follows that $|\mathrm{M}| \leq |C|$.

  – This property is called the *weak-duality* between the matching and vertex cover.

  – It implies that, in any graph, the size of maximum matching is at most the size of minimum vertex cover.

The König-Egeváry Theorem

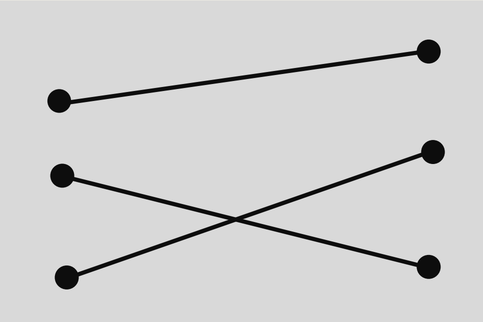
In bipartite graphs, the size of the ***maximum matching*** is equal to the size of the ***minimum vertex cover***.

Theorem 5.5 (König-Egeváry 1931).

In a bipartite graph, the size of ***maximum matching*** is equal to the size of ***minimum vertex cover***.

Proof.

- Let G be a bipartite graph with partite sets U and V.
- Let M be a *maximum matching* and
  C be a *minimum vertex cover* for G, respectively.
- It suffices to prove that $|M| \geq |C|$.



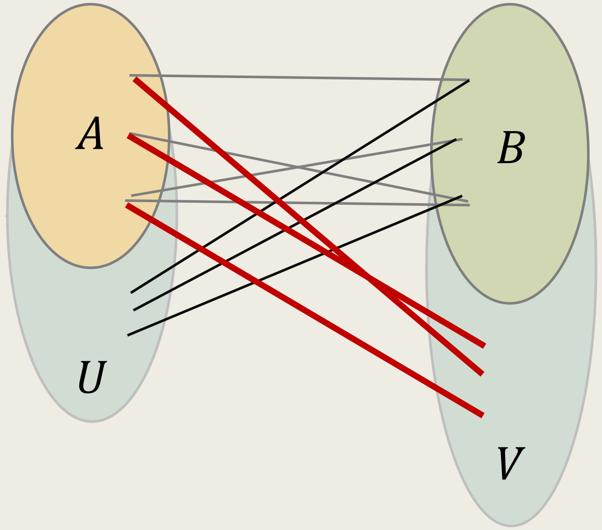
The matching M

Theorem 5.5 (König-Egeváry 1931).

In a bipartite graph, the size of ***maximum matching*** is equal to the size of ***minimum vertex cover***.

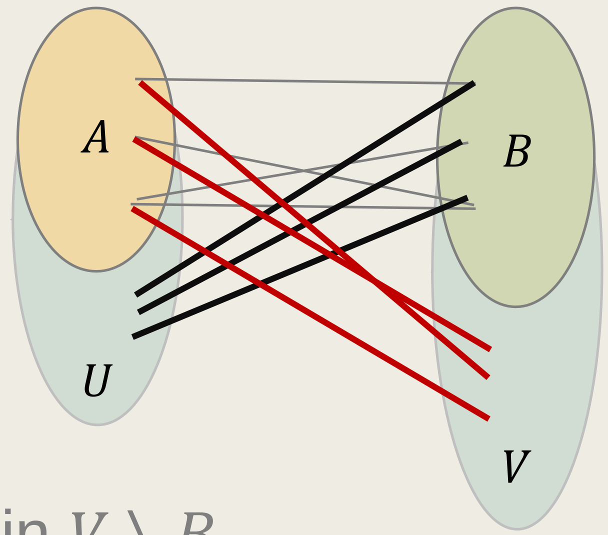
Proof.

- It suffices to prove that $|M| \geq |C|$.
  - Let $A \coloneqq U \cap C$ and $B \coloneqq V \cap C$.
  - We will prove that, there exists a matching M_A that matches all the vertices in A to the vertices in $V \setminus B$.

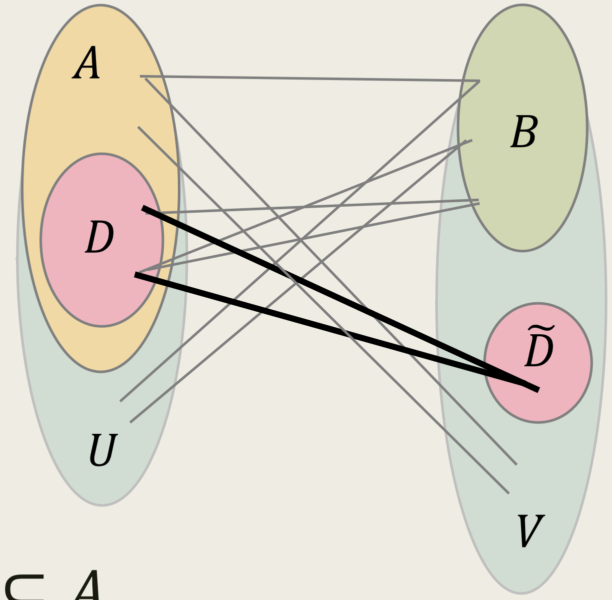


Proof.

- It suffices to prove that $|M| \geq |C|$.
- Let $A \coloneqq U \cap C$ and $B \coloneqq V \cap C$.
  - We will prove that, there exists a matching M_A that matches all the vertices in A to the vertices in $V \setminus B$.
  - **If the above is true**, then by a similar argument, there exists a matching M_B for B to $U \setminus A$.
  - The endpoints of the edges in $M_A \cup M_B$ are distinct.
    - So, $M_A \cup M_B$ is a matching of size $|A| + |B| = |C|$.
    - Hence, this will prove that $|M| \geq |A| + |B| = |C|$.



> It suffices to prove that, there exists a matching M_A that matches all the vertices in A to the vertices in $V \setminus B$.



- Suppose that there exists no such matching.
- Then, by Hall's matching theorem, there exists some $D \subseteq A$, such that

$$|\ N(D)\ \cap\ (V \setminus B)\ |\ <\ |D|.$$

  - Indeed, if $|N(D) \cap (V \setminus B)| \geq |D|$ holds for all $D \subseteq A$, then there exists a matching from A to $V \setminus B$.
  - Since there is no such matching, there must be such a $D \subseteq A$ with $|N(D) \cap (V \setminus B)| < |D|$.

It suffices to prove that, there exists a matching M_A that matches all the vertices in A to the vertices in $V \setminus B$.

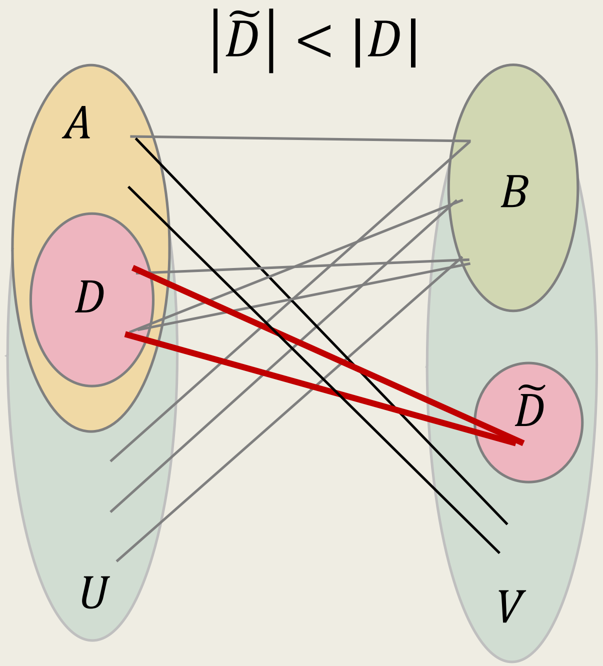
- If not, there exists some $D \subseteq A$, such that

  $$|\ N(D)\ \cap\ (V \setminus B)\ |\ <\ |D|\ .$$

- Let $\widetilde{D} \coloneqq N(D) \cap (V \setminus B)$, then $|\widetilde{D}| < |D|$.
- We claim that,

  $(A \setminus D) \cup \widetilde{D} \cup B$ is a valid vertex cover for G.

  – If this is true, we obtain a vertex cover with size smaller than $|A| + |B| = |C|$, a contradiction.



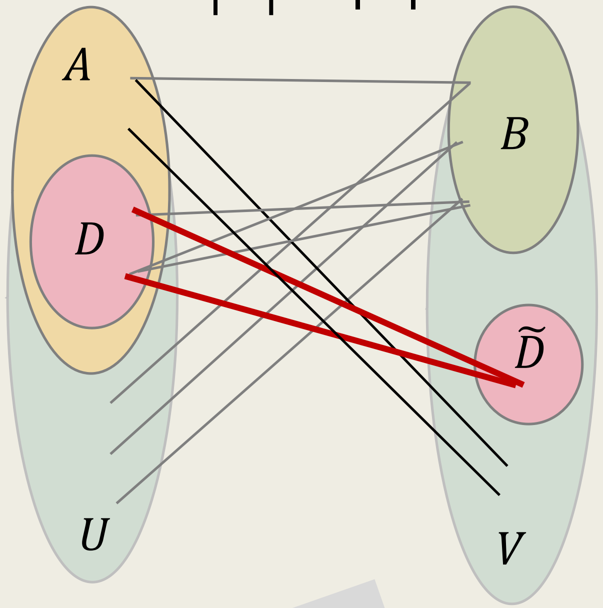
D is replaceable by $\widetilde{D}$.

It suffices to verify that,

$(A \setminus D) \cup \widetilde{D} \cup B$ is a valid vertex cover for G.

- Let $\widetilde{D} \coloneqq N(D) \cap (V \setminus B)$.
- There are four categories of edges in G.
  - $E_{A,B}$, $E_{U \setminus A,B}$ --- covered by B.
  - $E_{A \setminus D, V \setminus B}$ --- covered by $A \setminus D$.
  - $E_{D,\widetilde{D}}$ --- covered by $\widetilde{D}$.
- All the edges are covered.

$|\widetilde{D}| < |D|$



Since $C = A \cup B$ is a vertex cover,
there is not edge between $U \setminus A$ and $V \setminus B$.

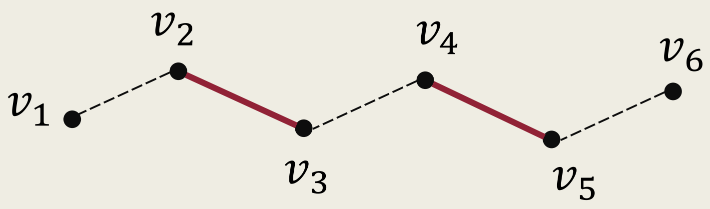
The Maximum Matching Problem

To compute a maximum-size matching for the input graph.

The Maximum Matching Problem

- Input :
  - A graph $G = (V, E)$.
- Output :
  - A matching $M \subseteq E$ that has the maximum size among all possible matchings.

Alternating Path & Augmenting Path

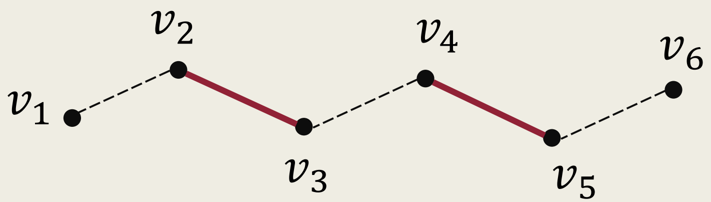


- Let M be a matching for a graph G.
  - An M-alternating path is a path that alternates between edges in M and edges not in M.
  - An **M-augmenting path** is an M-alternating path that both starts and ends at unmatched vertices.

Both v_1, v_2, v_3 and v_2, v_3, v_4, v_5 are M-alternating paths.

$v_1, v_2, v_3, v_4, v_5, v_6$ is an M-augmenting paths.

Observation



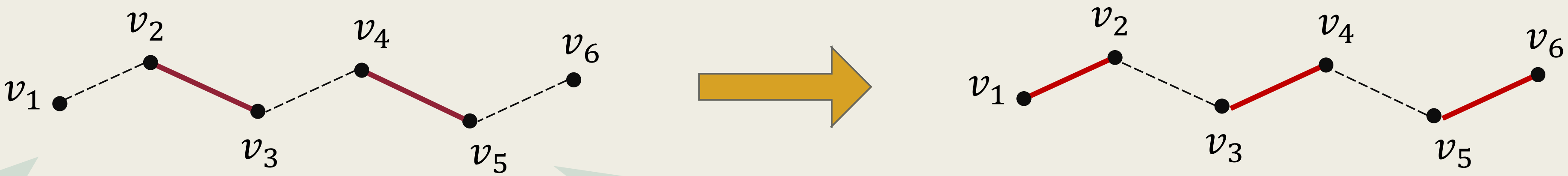
- We can see that,

  each M-augmenting path is a way to enlarge the size of M by 1.

  – This is done by swapping the status of the edges on the path.

    - Matched edges $\Longrightarrow$ *unmatched*
    - Unmatched edges $\Longrightarrow$ *matched*

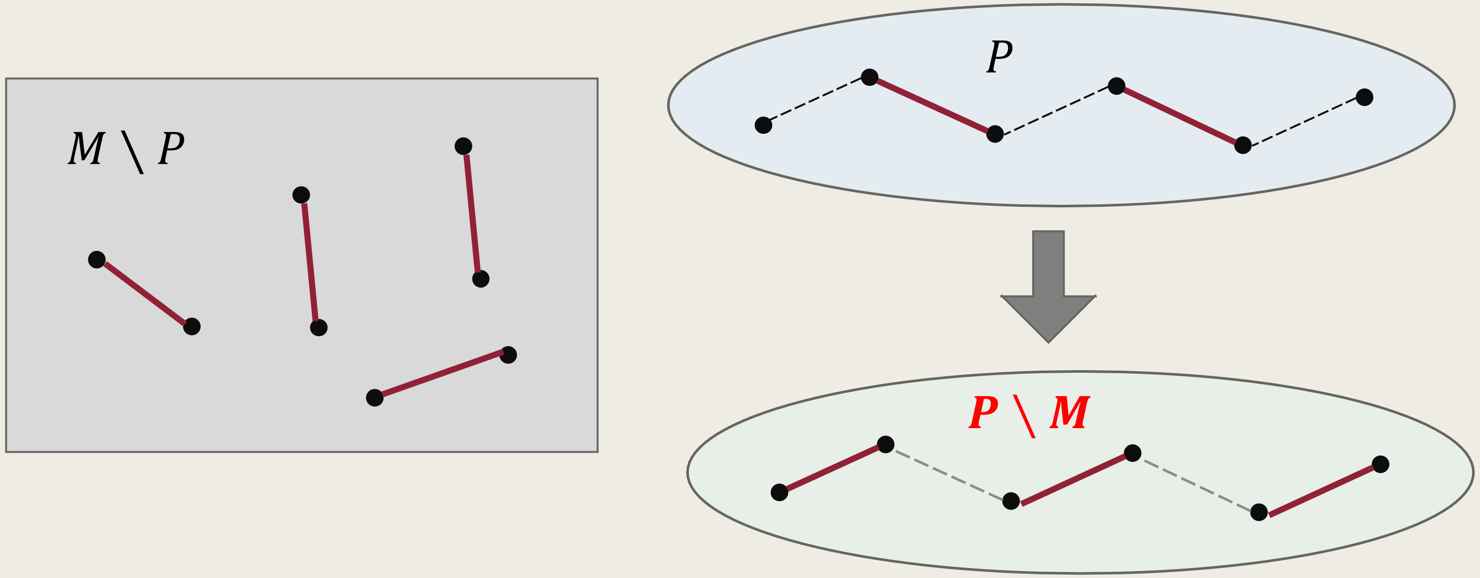
So, this is still a valid matching with size increased by 1.



v_1 and v_6 were unmatched.

All internal vertices are matched only by edges on the path.

Observation



- We can see that,

  each M-augmenting path P is a way to enlarge the size of M by 1.

- $M' \coloneqq (M \setminus P) \cup (P \setminus M)$ is a valid matching with $|M'| = |M| + 1$.

$M \Delta P$: the edges that appear exactly once in M and P.

A Simple Greedy Algorithm

- The observation suggests the following *greedy algorithm*.
  - Let $G = (V, E)$ be the input graph.

1. $M \leftarrow \emptyset$.
2. Repeat until there is no M-augmenting path in G.
   a. Compute an M-augmenting path P.
   b. Set $M \leftarrow (M \setminus P) \cup (P \setminus M)$.
3. Output M.

1. $M \longleftarrow \emptyset$.
2. Repeat until there is no M-augmenting path in G.
   a. Find an M-augmenting path P.
   b. Set $M \longleftarrow (M \setminus P) \cup (P \setminus M)$.
3. Output M.

The philosophy behind the algorithm is very simple :

"Make the current matching larger until no augmenting path exists."

- A direct question is that,

"Does it always output a maximum matching?"

Theorem 1. (Berge 1957).

A matching M in a graph G is a maximum matching if and only if G has no M-augmenting path.

- Theorem 1 assures the correctness of the greedy algorithm.
  - When there is no M-augmenting path, M is guaranteed to be maximum.

- We begin with some definition & helper lemma.

Symmetric Difference

- Let $G = (V, E)$ be a graph, and $A, B \subseteq E$ be two edge sets.
  - The symmetric difference of A and B is defined as

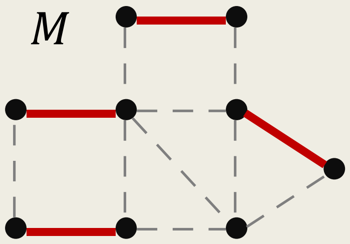
$$A \triangle B \coloneqq (A \setminus B) \cup (B \setminus A)\,.$$

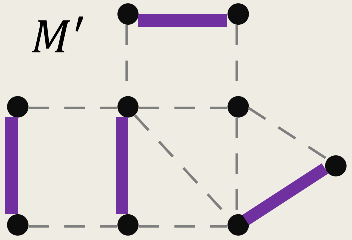
  - i.e., the set of edges that appear exactly once in A and B.

Lemma 2.

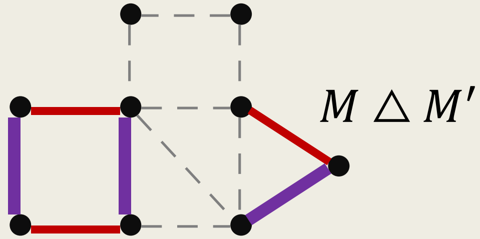
Let M, M' be matchings for a graph G. Then, every component of $M \triangle M'$ is a either *path* or a *cycle with an even length*.

- Let $F \coloneqq M \triangle M'$.
  - Each vertex in G is incident to at most two edges in F.
  - Hence, each component in F is either a path or a cycle.





- Consider any cycle in F.
  - The cycle alternates between edges in M and M'.
  - It must have an even length.



Theorem 1. (Berge 1957).

A matching M in a graph G is a maximum matching if and only if G has no M-augmenting path.

- Let us prove Theorem 1.
  - The direction ($\Longrightarrow$) is clear.
  - For the direction ($\Longleftarrow$), we prove the contrapositive statement.
    - We show that, if M' is a matching with $|M'| > |M|$, then G must have an M-augmenting path.

It suffices to prove that, if M' is a matching with $|M'| > |M|$, then G must have an M-augmenting path.

- Let $F \coloneqq M \triangle M'$.
  - By Lemma 2, F is a union of paths and even cycles.
- Since $|M'| > |M|$, there must be a component in F that has more edges from M' than M.
  - The component must be a path.
    Furthermore, it must start and ends with edges in M'.
  - The path is then an M-augmenting path.

The Maximum Matching Problem

- The Berge's theorem suggests the following simple algorithm.
  - Let $G = (V, E)$ be the input graph.

1. $M \leftarrow \emptyset$.
2. Repeat until there is no M-augmenting path in G.
   a. Compute an M-augmenting path P.
   b. Set $M \leftarrow (M \setminus P) \cup (P \setminus M)$.
3. Output M.

The Augmenting Path Problem

- To solve the maximum matching problem,
  it suffices to answer the *augmenting path problem*.

- Input :
  - A graph $G = (V, E)$ and a matching M for G.

- Goal :
  - Compute an M-augmenting path for G, or,
    Assert that there exists no such path.

The Augmenting Path Problem

- To solve the maximum matching problem,
  it suffices to answer the following *augmenting path problem*.

- In this lecture, we will introduce algorithms that solve the augmenting path problem.
  - $O(m)$ for bipartite graph.
  - $O(nm)$ for general graphs.

The Augmenting Path Problem

in Bipartite Graphs

For bipartite graphs,
the augmenting path problem can be solved by simple DFS in $O(n+m)$ time.

The Augmenting Path Problem in Bipartite Graphs

- Let $G = (V, E)$ be a bipartite graph *with partite sets* $\boldsymbol{A}$ ***and*** $\boldsymbol{B}$, and $\boldsymbol{M}$ be a matching for G.

- We introduce an algorithm that computes in $O(m)$ time either
  - An M-augmenting path for G, or,
  - A vertex cover C for G with $|C| = |M|$.

  Note that, in the latter case, M is a maximum matching by the weak duality, and hence no augmenting path exists.

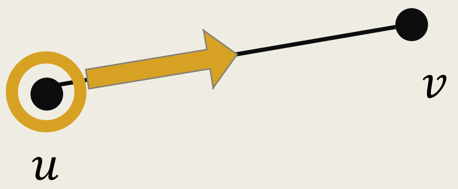
An Augmenting Path Algorithm for Bipartite Graphs

- Let $G = (V, E)$ be a bipartite graph *with partite sets* $\boldsymbol{A}$ ***and*** $\boldsymbol{B}$, and $\boldsymbol{M}$ be a matching for G.

- The algorithm attempts to compute an M-augmenting path
  starting at an unmatched vertex in $\boldsymbol{A}$
  using a DFS-based recursive procedure ***aug-path()***.
  - If it succeeds for some unmatched vertex $v \in \boldsymbol{A}$, then we're done.
  - If it fails for *every unmatched vertex* in $\boldsymbol{A}$, then a vertex cover C with $|C| = |M|$ can be defined.

The DFS-based Recursive Procedure ***aug-path()***

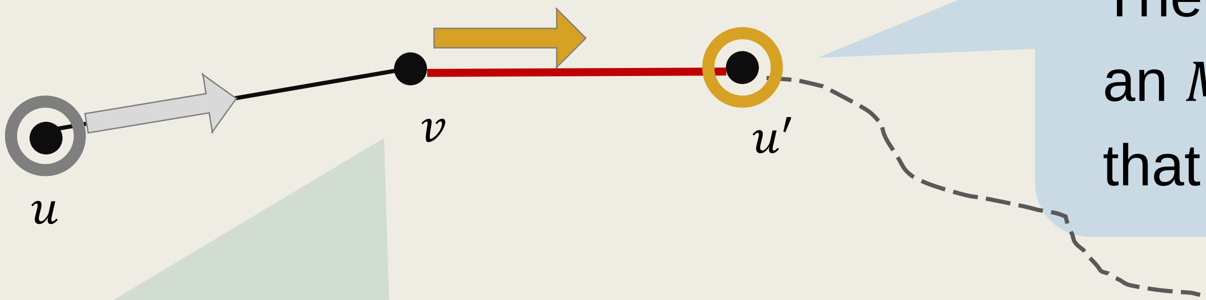
- Finding an augmenting path in a bipartite graph can be handled by a simple & intuitive DFS-based procedure.
  - We start with an unmatched vertex, say, u.
    - The goal is to find an M-augmenting path starting from u.
  - Consider ***each neighbor*** *of* u, say, v.



If v is <u>*unmatched*</u>,
then u, v is an M-augmenting path,
and we're done.

- We start with an unmatched vertex, say, u.
  - Our goal is to find an M-augmenting path starting from u.

- Consider each neighbor of u, say, v.



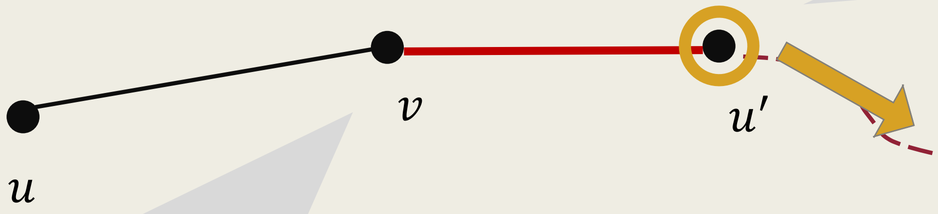
Then, the goal becomes finding an M-augmenting path that starts that ***starts from*** u'.

If v is ***matched***, then
to form an M-augmenting path that passes v,
we must follow the matched edge to some u'.

This is a recursive problem that starts at the vertex u'.

- We start with an unmatched vertex, say, u.
  - Our goal is to find an M-augmenting path starting from u.

- Consider each neighbor of u, say, v.



Then, the goal becomes finding an M-augmenting path that starts that ***starts from*** u'.

This is a recursive problem that starts at the vertex u'.

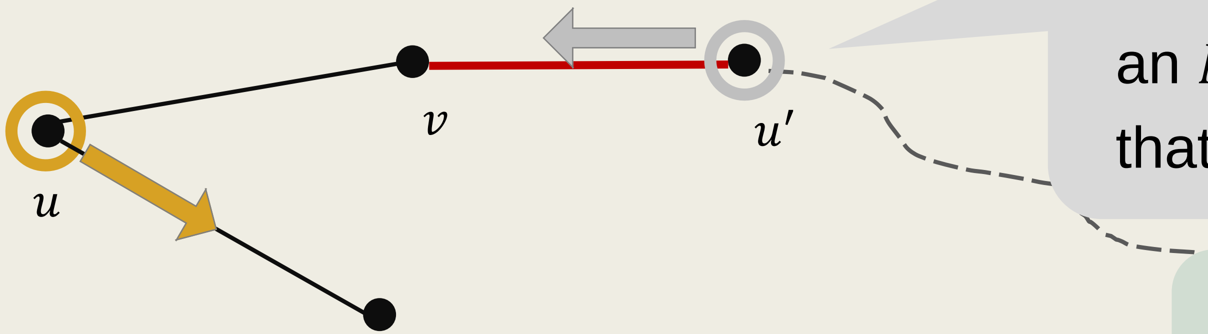
If v is ***matched***, then
to form an M-augmenting path that passes v,
we must follow the matched edge to some u'.

If the recursion succeeds,
we have an augmenting path for u.

– We start with an unmatched vertex, say, u.

  ▪ Our goal is to find an M-augmenting path starting from u.

– Consider each neighbor of u, say, v.



Then, the goal becomes finding an M-augmenting path that starts that ***starts from*** u'.

This is a recursive problem that starts at the vertex u'.

If it fails, then we go back to u,
and continue to examine the next neighbor
until all its neighbors have been examined.

The DFS-based Recursive Procedure ***aug-path()***

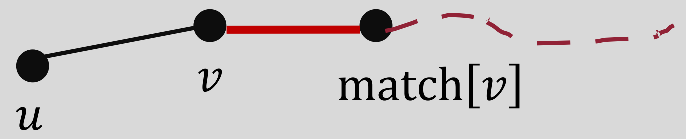
- To formally describe the procedure,
  let's assume the following.
  - Each vertex in G is associated with a status, which is either *<u>visited</u>* *or* *<u>unvisited</u>*.
  - For each vertex v, let $\mathrm{match}[v]$ denote the vertex to which v is matched.
    - $\mathrm{match}[v] = -1$ if v is unmatched.

- The DFS-based recursive procedure goes as follows.

<u>Procedure</u> Aug-Path(u)

1. Mark u as *visited*.
2. For each neighbor v of u, do.
   - If v is *unmatched*, then return the path $\{u, v\}$.
   - If match[v] is *unvisited* and (P ←Aug-Path(match[v])) ≠ ∅, then return the path $\{u, v, P\}$.
3. Return ∅.

Augmenting path from $\mathrm{match}[v]$ is found.



An Augmenting Path Algorithm for Bipartite Graphs

- Let $G = (V, E)$ be the input bipartite graph with partite sets A and B, and M be a matching for G.

<u>An Augmenting Path Algorithm</u> (for Bipartite Graphs).

1. Mark all the vertices as *unvisited*.
2. For each *unmatched* vertex $u \in A$, do
   - If ($P \leftarrow$ Aug-Path(u)) $\neq \emptyset$, then return P.
3. Report “No” and return a <u>*vertex cover*</u> C with $|C| = |M|$.

We will show
how this can be done.

Analysis of the Algorithm

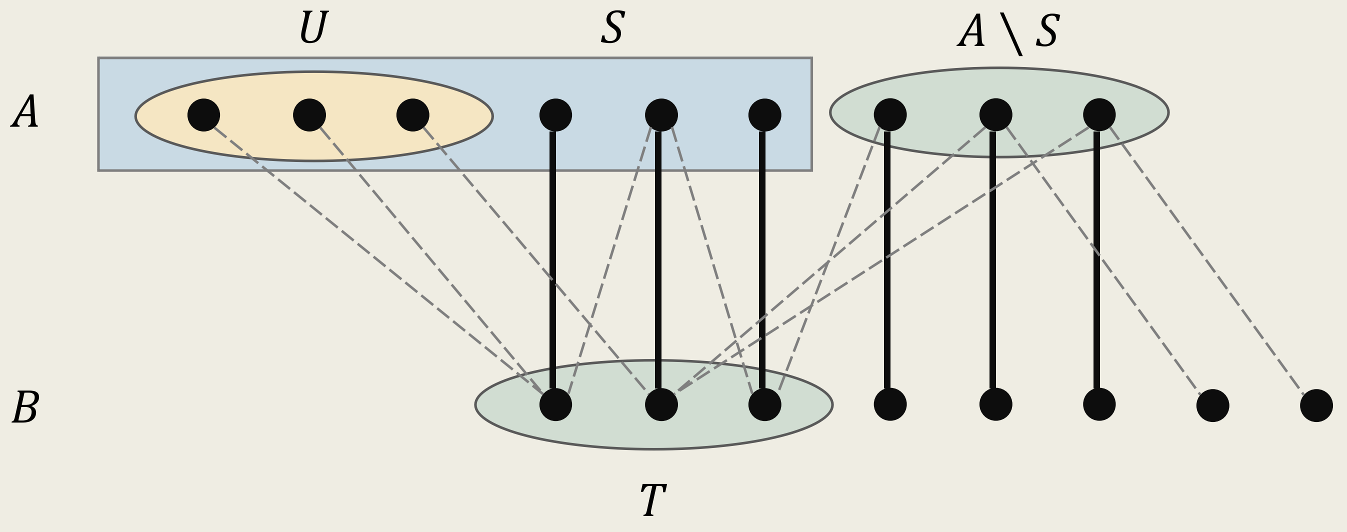
- Since each vertex is visited at most once and each edge is examined at most twice by the procedure Aug-Path(),
  - The algorithm runs in $O(n+m)$ time.

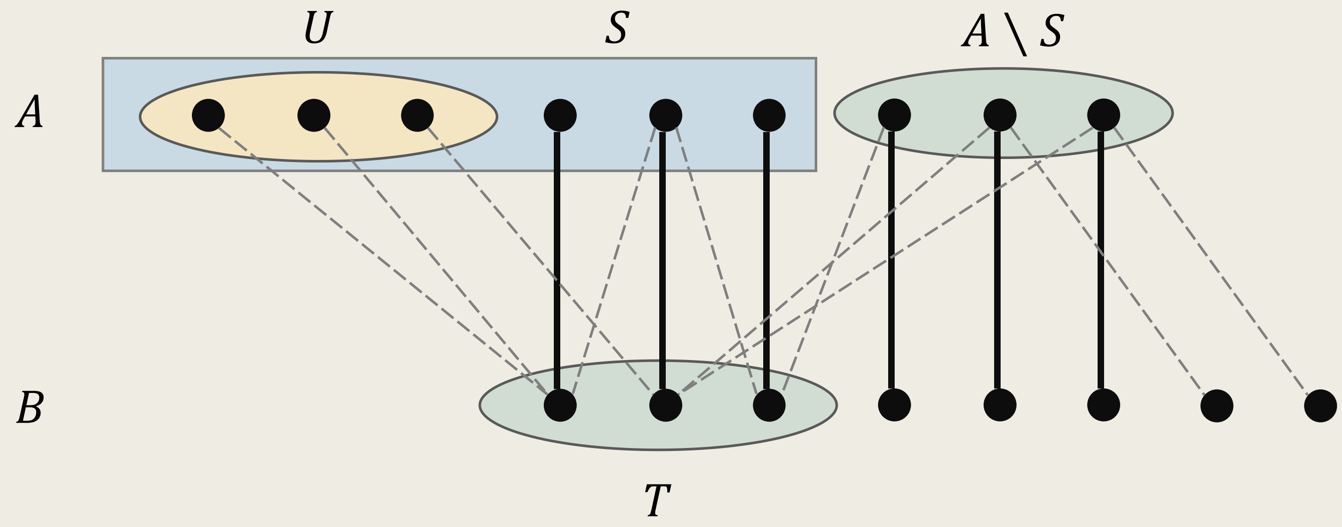
- It is clear that, if Aug-Path(u) returns a non-empty path P, then an M-augmenting path starting at u is found.

- To prove the correctness of the algorithm,
  we need to prove that,
  - There exists no M-augmenting path in the graph when the algorithm reports “No.”

Notations

- Let A and B be the two partite sets of G.
  - Let U be the set of unmatched vertices in A.
  - Let S be the vertices in A that are marked as *visited*.
  - Let T be the set of vertices in B that are matched to $S \setminus U$ by M.





Theorem 3.

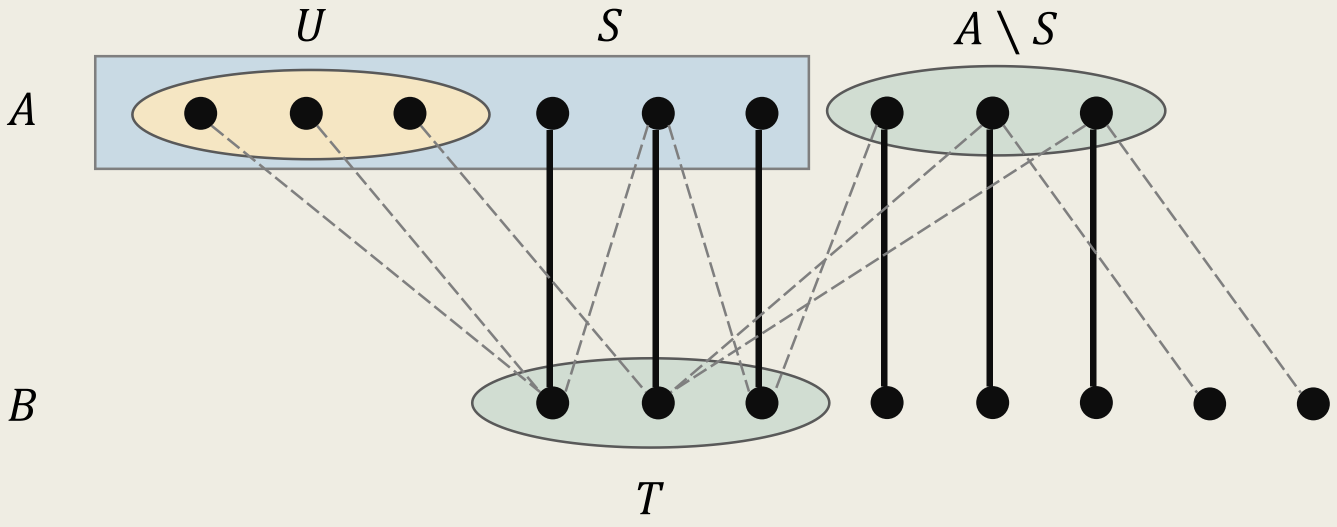
If the Augmenting Path Algorithm reports “No,” then the set $C \coloneqq (A \setminus S) \cup T$ is a vertex cover for G with size M.

Note that, this is also a *constructive proof* for the König-Egeváry theorem.

Observation 1.

Since v is marked visited,
it is visited by a recursion call that
originates from some $u \in U$.

- For any $v \in S \setminus U$,
  - There is an M-alternating path that starts at some $u \in U$ and ends at v with a matched edge in M.

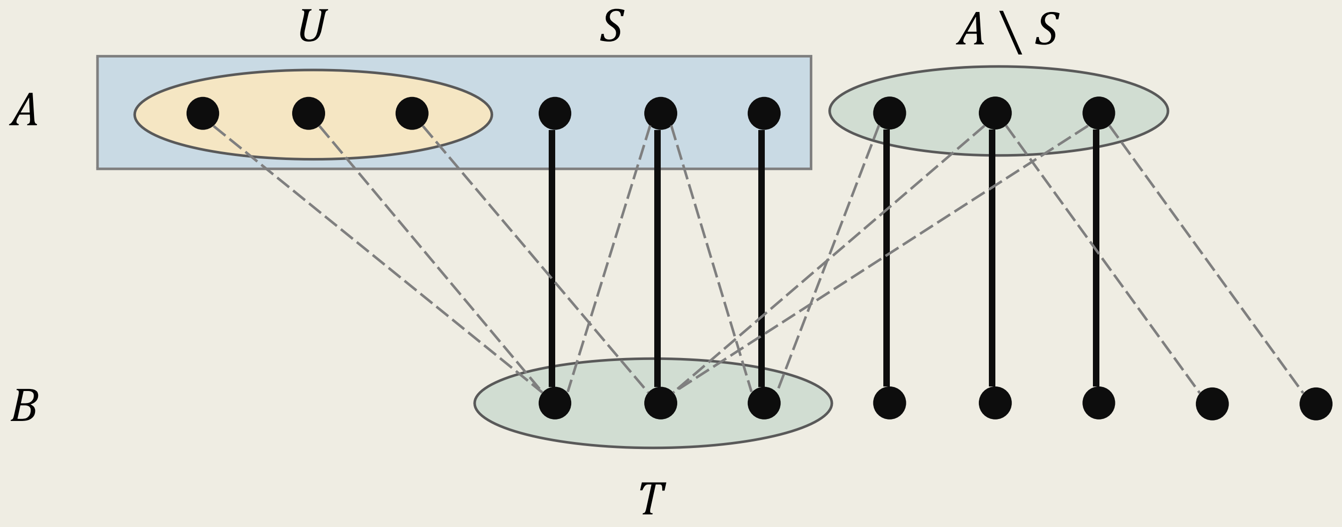


Observation 2.

- There exists no edge between S and $B \setminus T$.
  - Vertices in $A \setminus S$ are unvisited.

  Hence, there exists no edge between S and the matched vertices in $B \setminus T$.

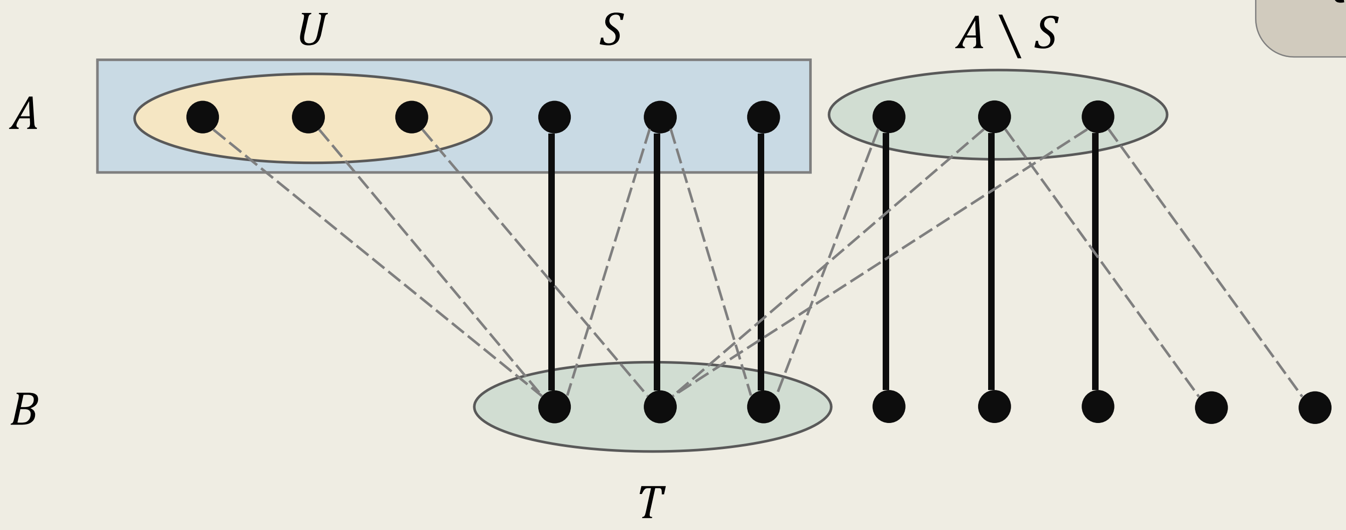
Otherwise, that matched vertex should be in T.



Observation 2.

- There exists no edge between S and $B \setminus T$.
  - If there exists an edge between S and some unmatched vertex in B, it will form an augmenting path that will be found by the recursive procedure.

A contradiction since the algorithm reports “No.”



Theorem 3.

If the Augmenting Path Algorithm reports "No," then the set $C := (A \setminus S) \cup T$ is a vertex cover for G with size M.

- The edges between S and T can be covered by T.
- By Observation 2, the remaining edges can be covered by $A \setminus S$.
- Hence, C is a vertex cover for G.