Matching algorithms (Cont…)

RAMESH K. JALLU

LEC-06

DT. 20/01//22

Basic definitions (Recap)

- Given a graph $G = (V, E)$ and a matching M of G
  1. A ***stem*** with respect to M is an alternating path of even length from an unmatched vertex v (called the ***root*** of the stem)
     - Observation: The last edge on the stem belongs to M
  2. A ***blossom*** B with respect to M is a
     I. a cycle in G consisting of $2k + 1$ edges
     II. exactly k edges of the cycle belong to M, and
     III. one of the vertices v of the cycle (we call it the ***base***) is such that there exists a stem from an unmatched vertex w to v
        - Observation: The two edges of the blossom touching the base are not in M. Other than that, every second edge on the blossom belongs to M

Modifying the existing algorithm

- We have to modify our existing algorithm for bipartite graphs so that

1. It can detect existing blossoms, and
2. It can go on finding augmenting paths even in the presence of blossoms

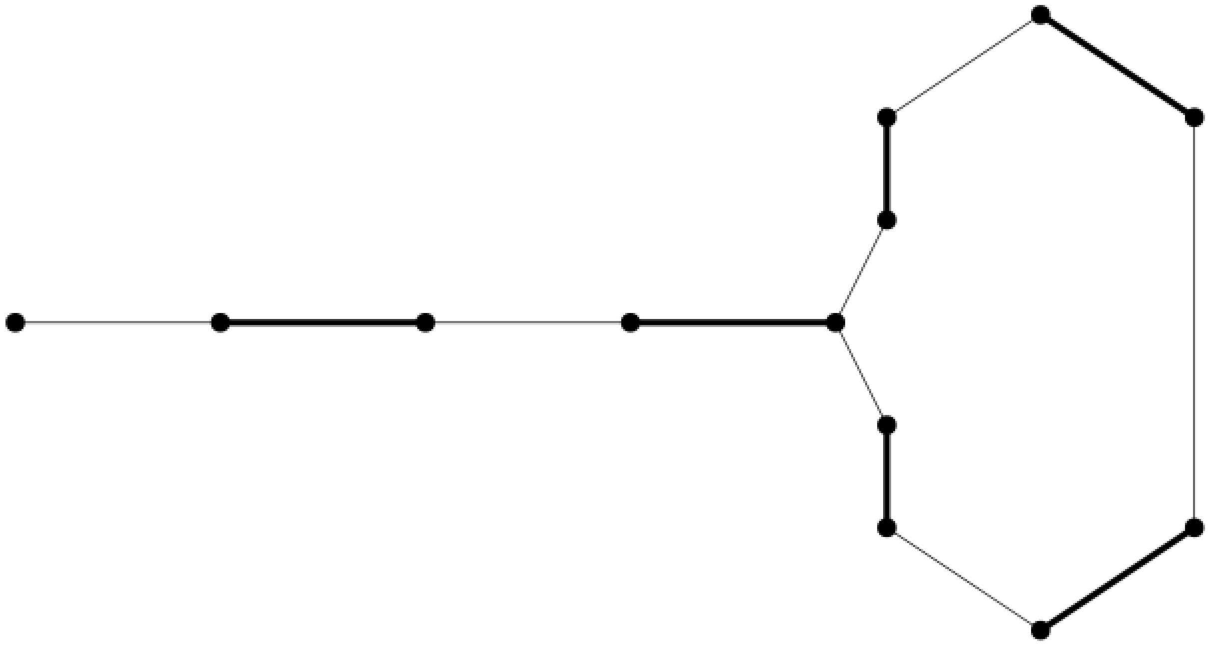
Modifying the existing algorithm

- We have to modify our existing algorithm for bipartite graphs so that
  1. It can detect existing blossoms, and How?
  2. It can go on finding augmenting paths even in the presence of blossoms

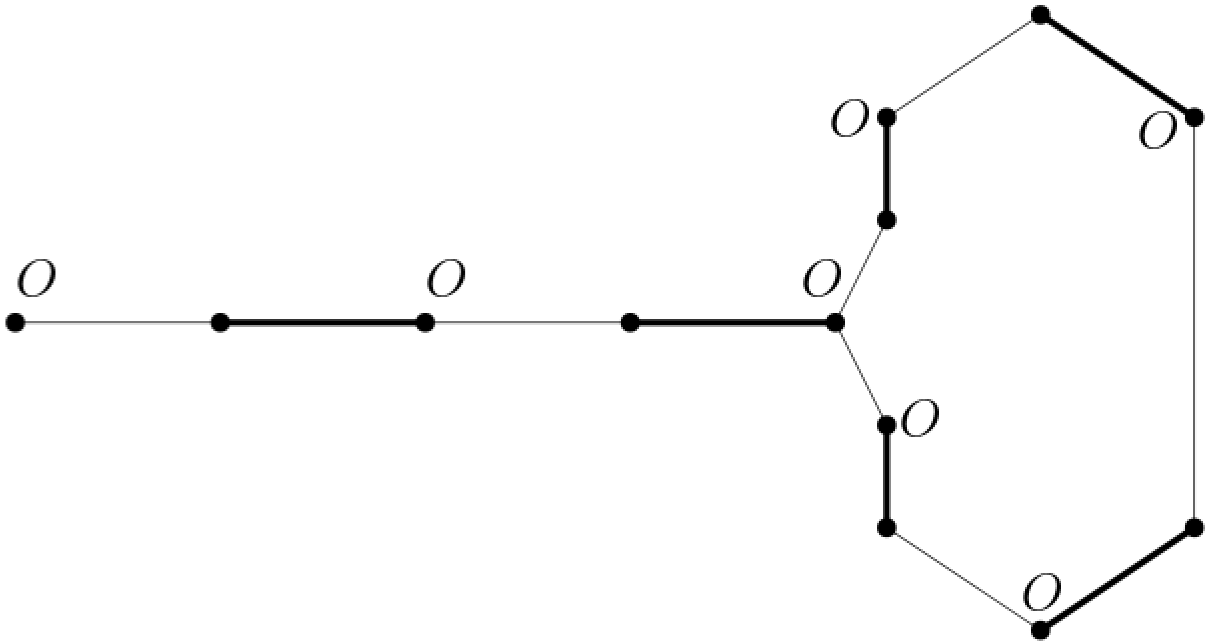
Detecting blossoms

- Perform alternating path search of the bipartite matching algorithm: start from an unmatched vertex
  1. Label vertices at even distances from the root as ***outer***
  2. Label vertices at odd distances from the root as ***inner***
- If two outer vertices found to be adjacent in the graph, we have a blossom

Example



Example

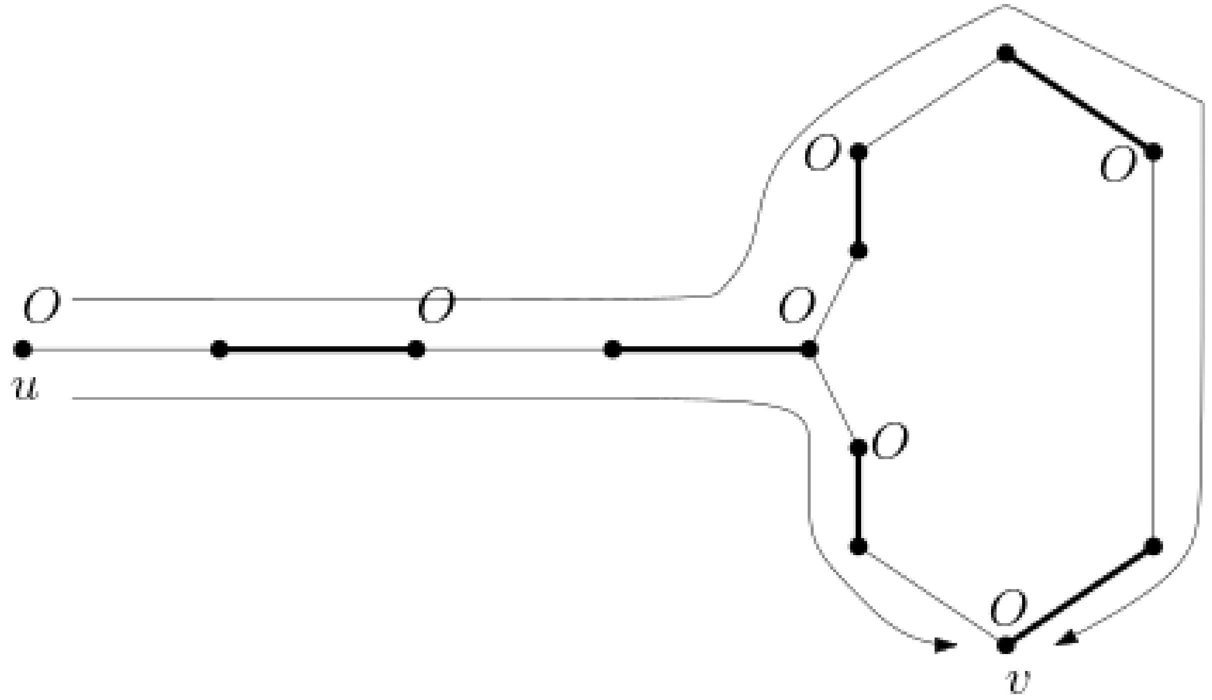


Algorithm idea

- To find an augmenting path in general graph, we will modify the procedure for bipartite graphs, so that it also detects blossoms
- If blossoms found, we shrink the blossom and restart on the new graph
- Any augmenting path found in new graph can easily be translated to an augmenting path in the original graph
- We will prove that, if the matching is maximum in the new graph, then it is also maximum in the original graph

Key observation 1

- Let M be a matching of G and B be a blossom. While searching for an augmenting path from an unmatched node u in G, if we discover B, then there is an alternating path from u to any node on B ending with a matched edge



Key observation 2

- Let M be a matching of G and B be a blossom. Consider the graph G' obtained by contracting B to a single vertex. Then the matching M' of G' induced by M is maximum in G' if and only if M is maximum in G
- *Proof* : If M' is not maximum in G', then G' contains an augmenting path P' wrt M'
- ***Case 1***: If P' doesn't have any vertex common in B, then P' is also an augmenting path in G. Contradiction to fact that M is maximum
- ***Case 2***: If P' intersects B, then some portion of the blossom is in interior of P' in G'
- Let P' enters B at a vertex v, and let u be the vertex where P' leaves B
- If $u = v$, then u is nothing but the base of B
- If $u \neq v$, then consider the augmented path in G' up to v and one of the alternating path from v to u in B and the rest of the path of P'
- The path P' is then an augmenting path in G. A contradiction

Proof (Cont⋯)

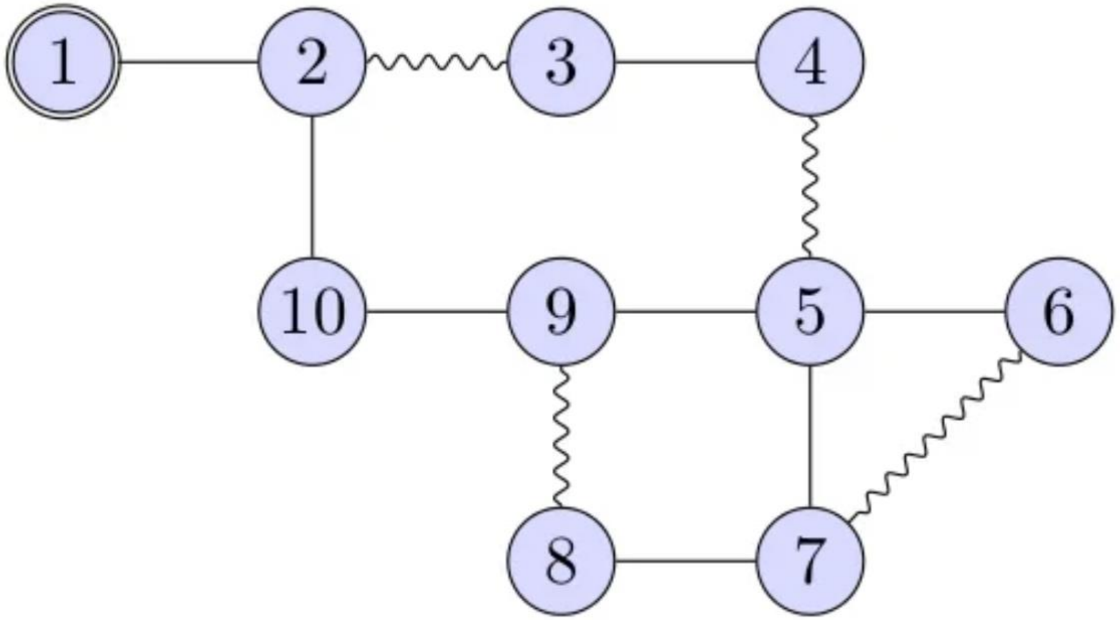
- Now assume that M is not a maximum matching in G
- We will show that M' is not a maximum matching in G'
- Let P be an augmenting path in G (Observe P is of odd length)
- P must intersect the blossom (otherwise P is also an augmenting path in G')
- P must pass through the stem of B
- If there are blossom's edges included in the augmenting path, contracting the blossom will eliminate an even number edges
- Thus, with an even number of alternating edges removed, the path in G remains an odd-length augmenting path in G'
- We arrived at a contradiction as P' is an augmenting path in G'

Edmond's algorithm

1. Start with an empty matching M
2. For all unmatched vertices $v \in V$
   a. Search for simple alternate path P from v
      - Shrink any blossom found
   b. If the path P ends at an unmatched vertex
      - Update M
   c. Else if no augmenting path found then
      - Ignore v for future searches
3. Return M

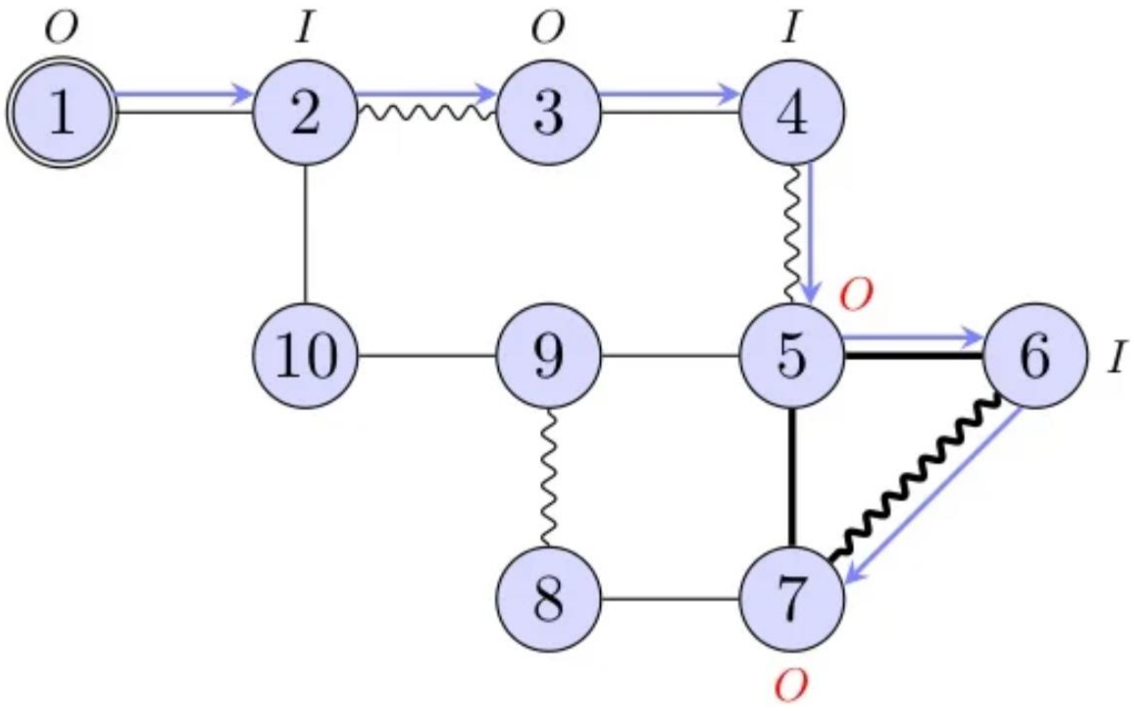
Example

$|M| = 4$



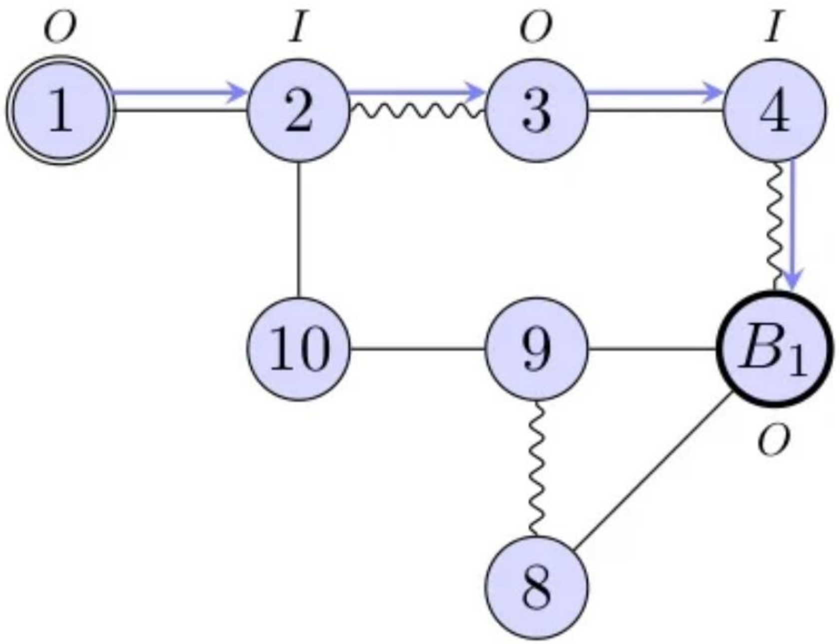
Example

$|M| = 4$



Example

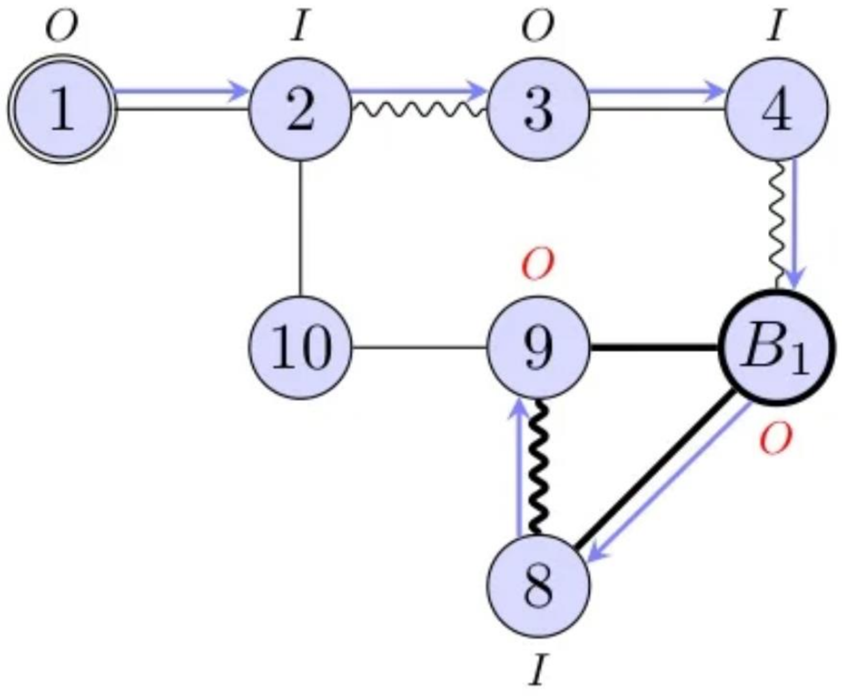
$|M| = 4$



$B_1 = 5, 6, 7$

Example

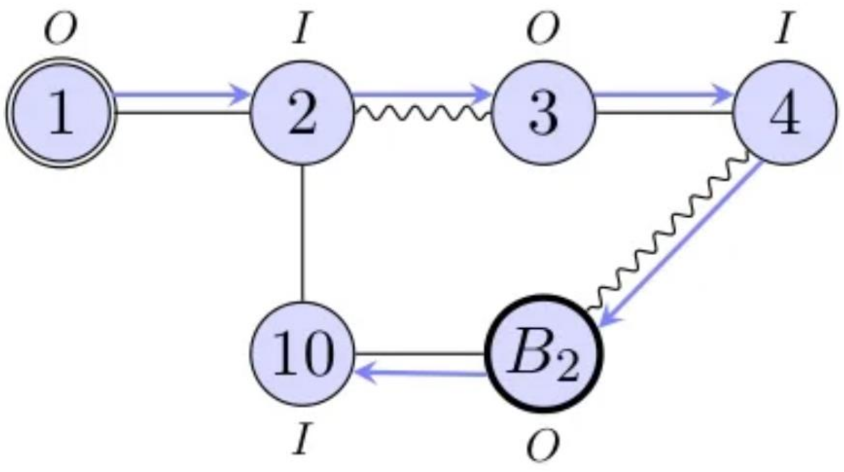
$|M| = 4$



$B_1 = 5, 6, 7$

Example

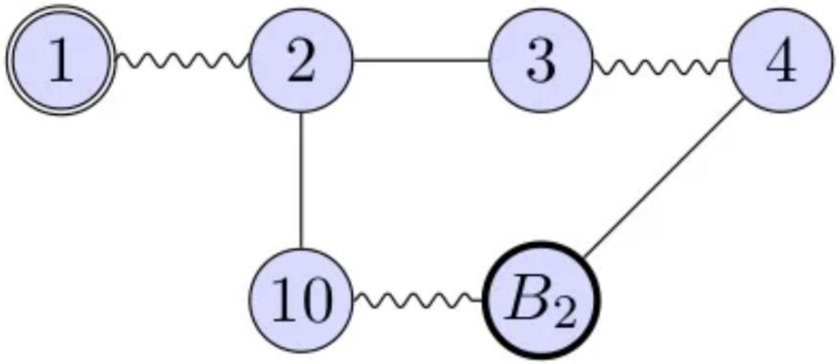
$|M| = 4$



$B_1 = 5, 6, 7$
$B_2 = B_1, 8, 9 = 5, 6, 7, 8, 9$

Example

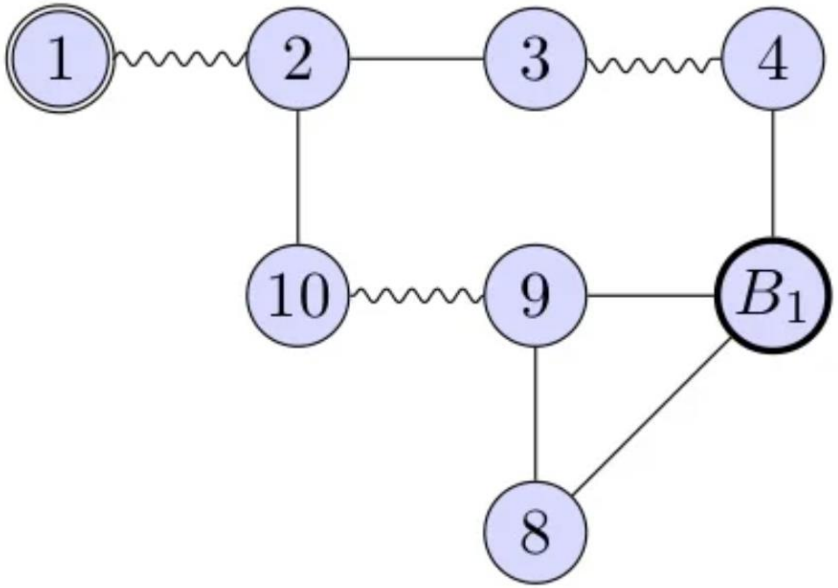
$|M| = 4$



$B_1 = 5, 6, 7$
$B_2 = B_1, 8, 9 = 5, 6, 7, 8, 9$

Example

$|M| = 4$

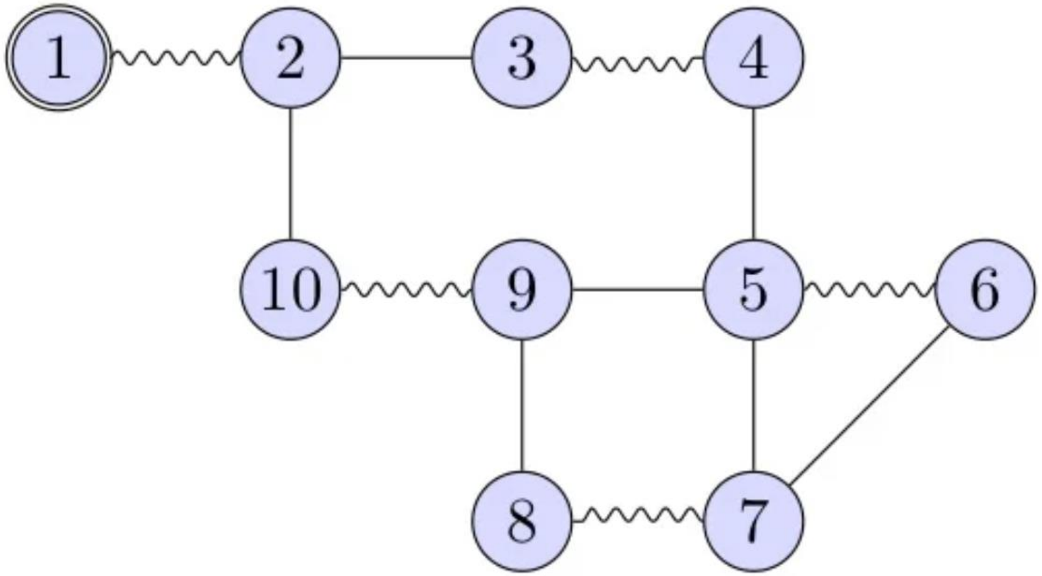


$B_1 = 5, 6, 7$
$B_2 = B_1, 8, 9 = 5, 6, 7, 8, 9$

Example

$|M| = 4$

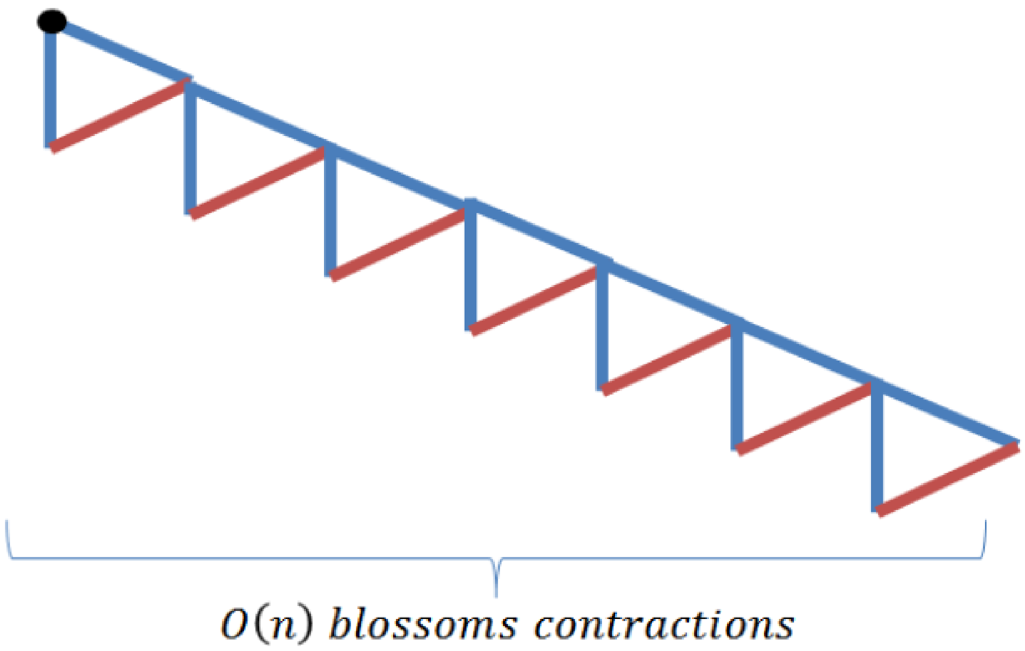
$|M| = 5$



$B_1 = 5, 6, 7$
$B_2 = B_1, 8, 9 = 5, 6, 7, 8, 9$

Lemma

- Each augmenting path iteration can have $O(n)$ Blossom contraction steps
- ***Proof***: One contraction of a blossom reduces the nodes by at least two
- Therefore, we can have at most $n/2$ blossom contractions
- As a result, each augmenting path iteration cannot have more than $O(n)$ blossom contraction steps



Complexity

- Contraction of a blossom takes $O(n + m)$ time
- An augmenting path can have at most $O(n)$ blossoms
- Total cost = No. of iterations•(time spent per iteration)

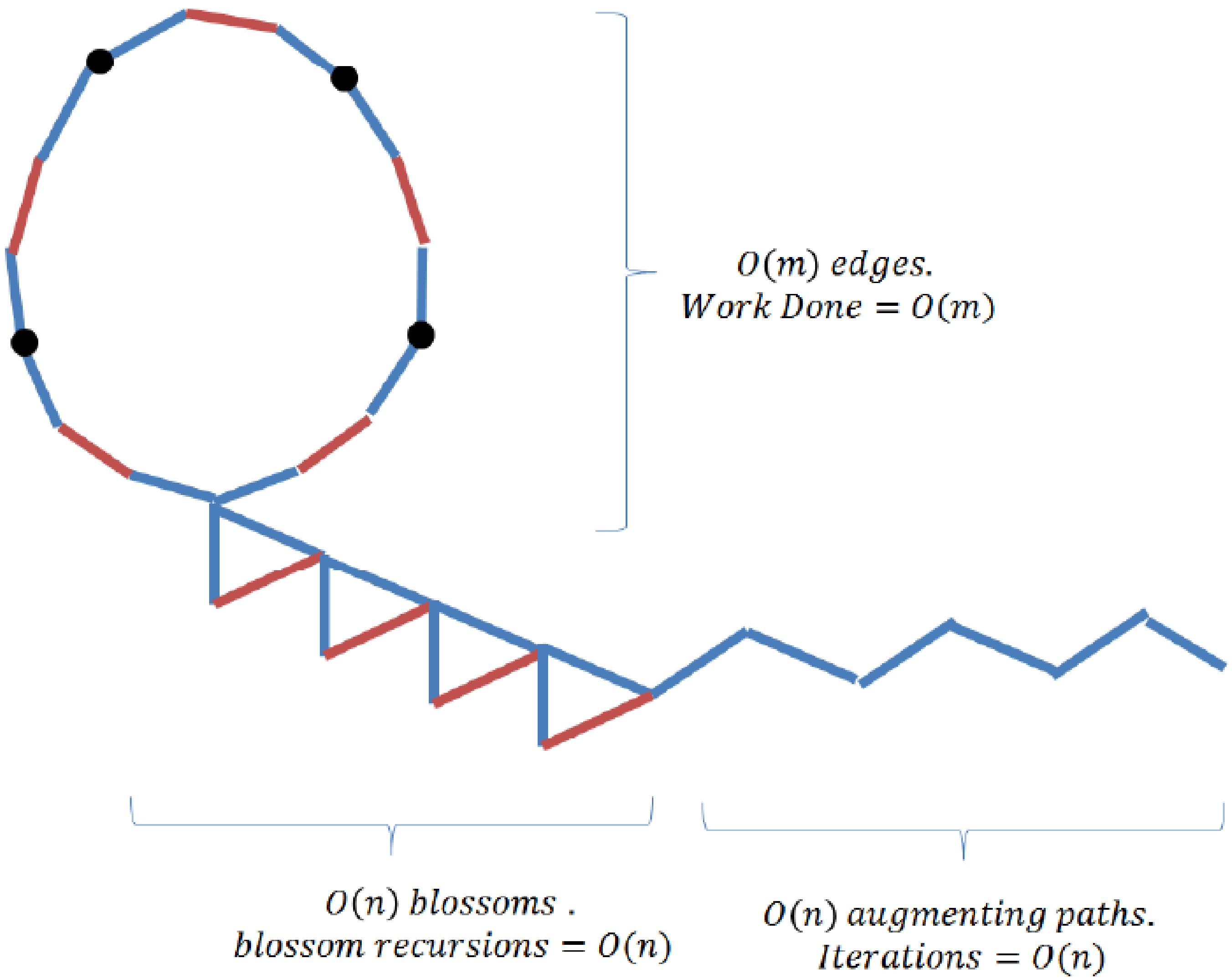
$= O(n)$•(time spent for case a + case b + case c)

$= O(n) \cdot (O(m) + (O(m) + O(m)) \cdot O(n))$

$= O(n^2 m)$

1. Start with an empty matching M
2. For all unmatched vertices $v \in V$
   a. Search for simple alternate path P from v
      - Shrink any blossom found
   b. If the path P ends at an unmatched vertex
      - Update M
   c. Else if no augmenting path found then
      - Ignore v for future searches
3. Return M

Tight analysis



- Thank you!