Maximum Matching in Graphs

Omar Ghaleb

OUTLINE

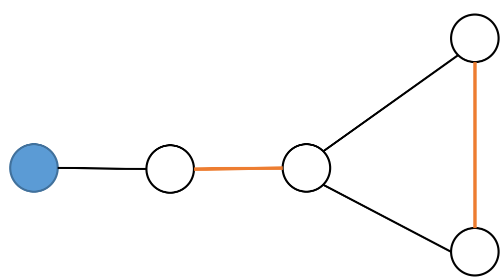
- Introduction
- Bipartite Graphs
- General Graphs
- Conclusion

INTRODUCTION

- $G = (V, E)$ - undirected graph with $|V| = n, |E| = m$, where n, m are number of vertices and edges respectively.
- M is a matching in G if it is a subset of E such that no two adjacent edges share a vertex.
- We say that the matching is maximum if we cannot find a better M has more edges.
- Maximum matching is not unique.

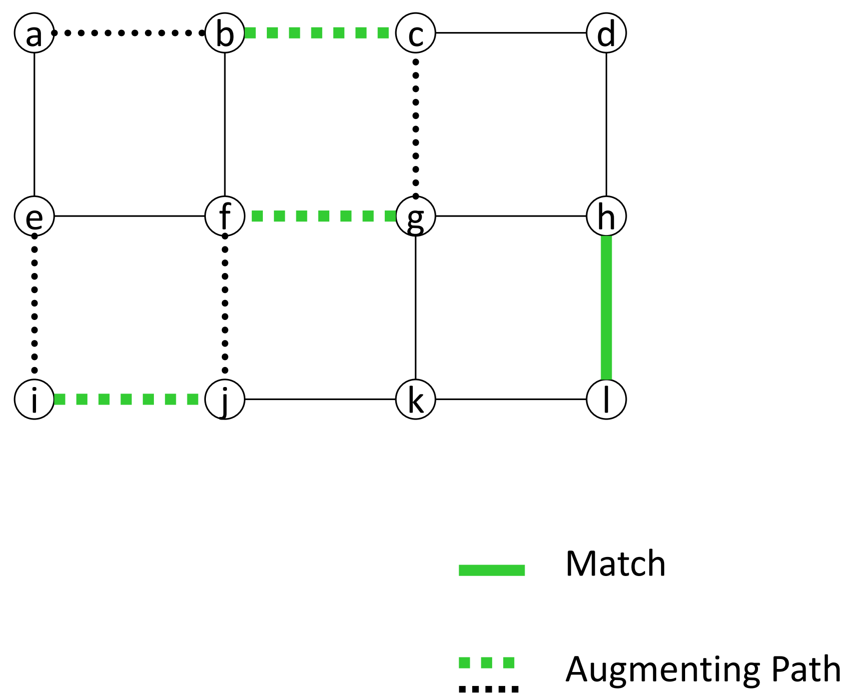
INTRODUCTION

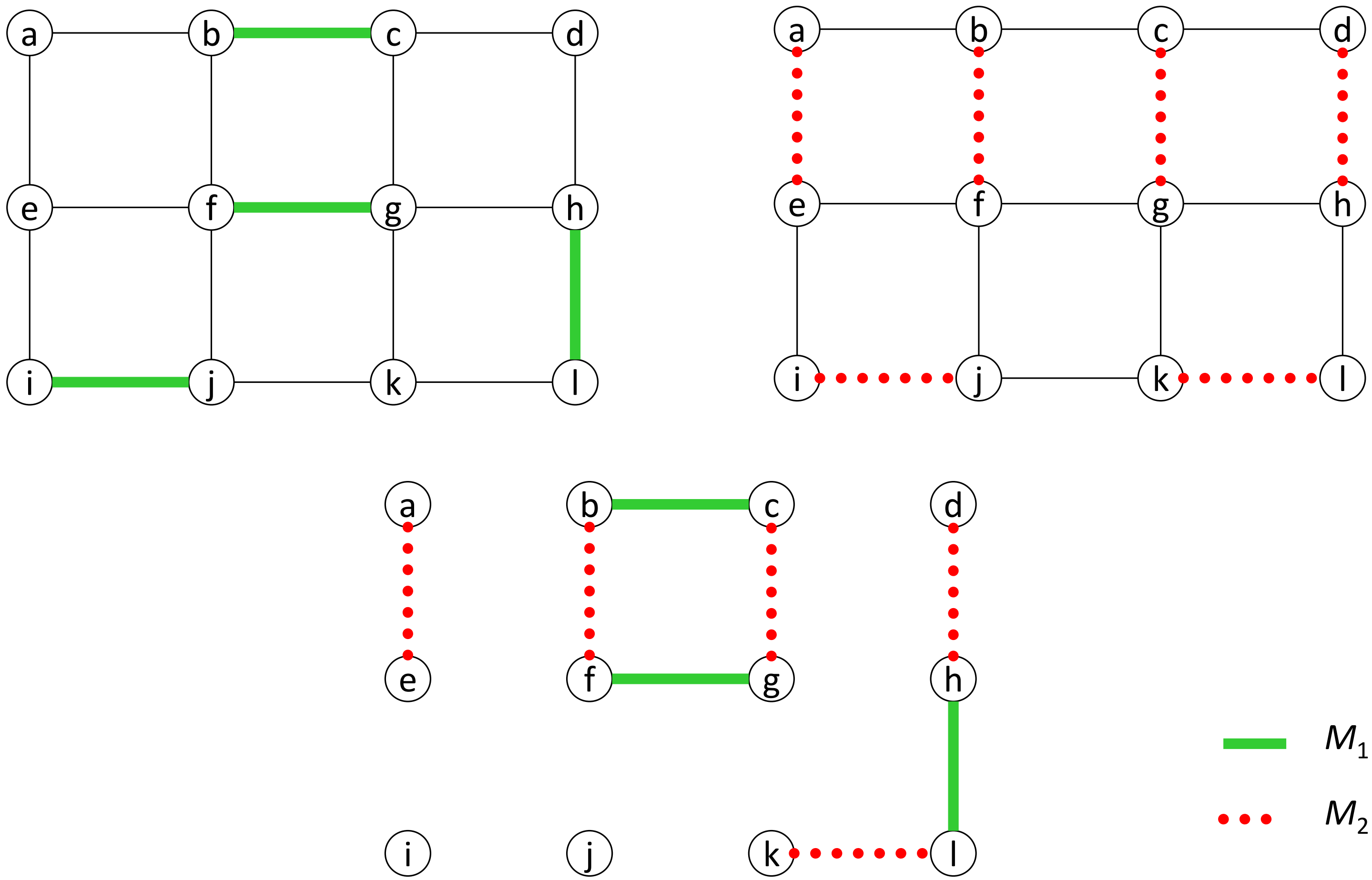
- A vertex is exposed if its not in any matching.



Augmenting Paths

- Alternating Path:
  - Is a path whose edges are alternating between being in M and not being in M.
- Augmenting Path:
  - Is a path found from an exposed vertex that can add a matching to M.

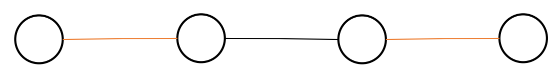

a
b
c
d
e
f
g
h
i
j
k
l
Match
Augmenting Path



$M_1 \oplus M_2$

Berge’s Theorem

- A matched graph (G,M) has an augmenting path IFF M is not a maximum.



Bipartite Graphs

- A graph G who can be divided into two sets, each one contains set of vertices A, B and each edge connects a vertex from A to a B.
- The bipartite graph has no cycles with odd number of edges.

- Ford-Fulkerson algorithm $O(nm)$
- Hopcroft and Karp Algorithm $O(\sqrt{n}m)$

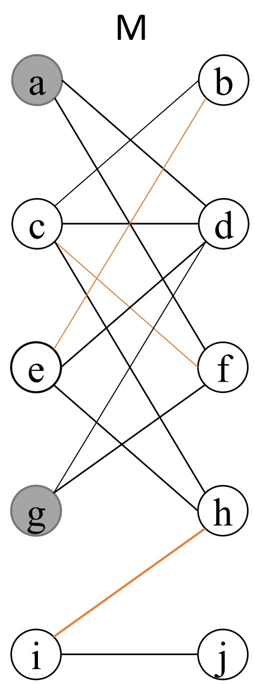
Algorithm for Bipartite

- Hopcroft and Karp Algorithm
- Have any matching M
- If there exists an augmenting path corresponding to M:
  - Find the path P
  - Add it to the matching M using symmetric difference $M' = M \oplus \mathrm{P}$
  - Then let $M = M'$

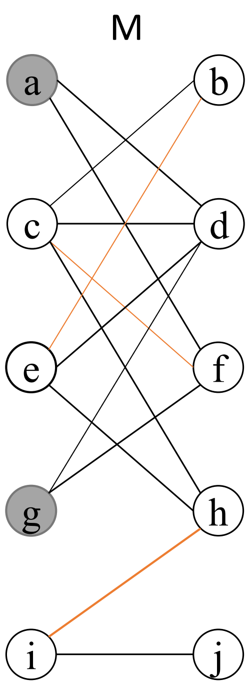
  If no augmenting paths found then we have a maximum matching in the bipartite graph

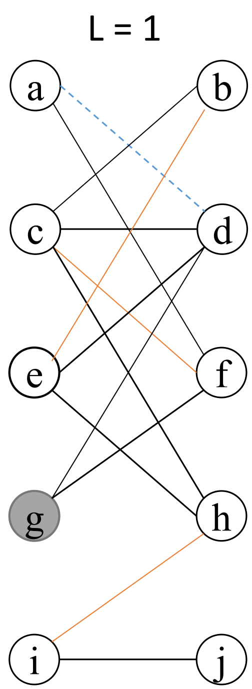
  - This algorithm runs in $O(\sqrt{n}m)$-best known deterministic algorithm for bipartite.
  - $(\sqrt{n})$ $phases\ required\ to\ find\ all\ augmenting\ paths$.

Example:

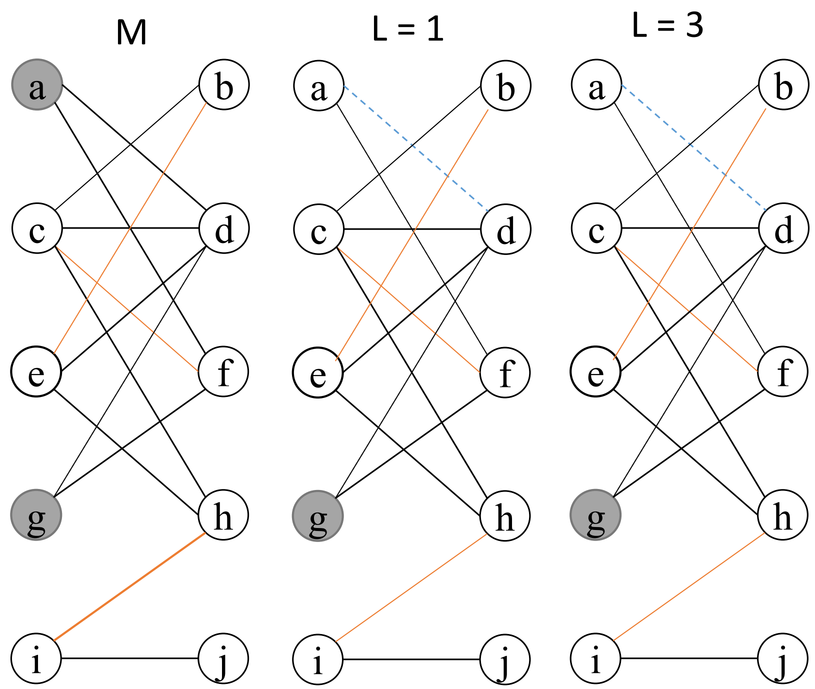


Example:

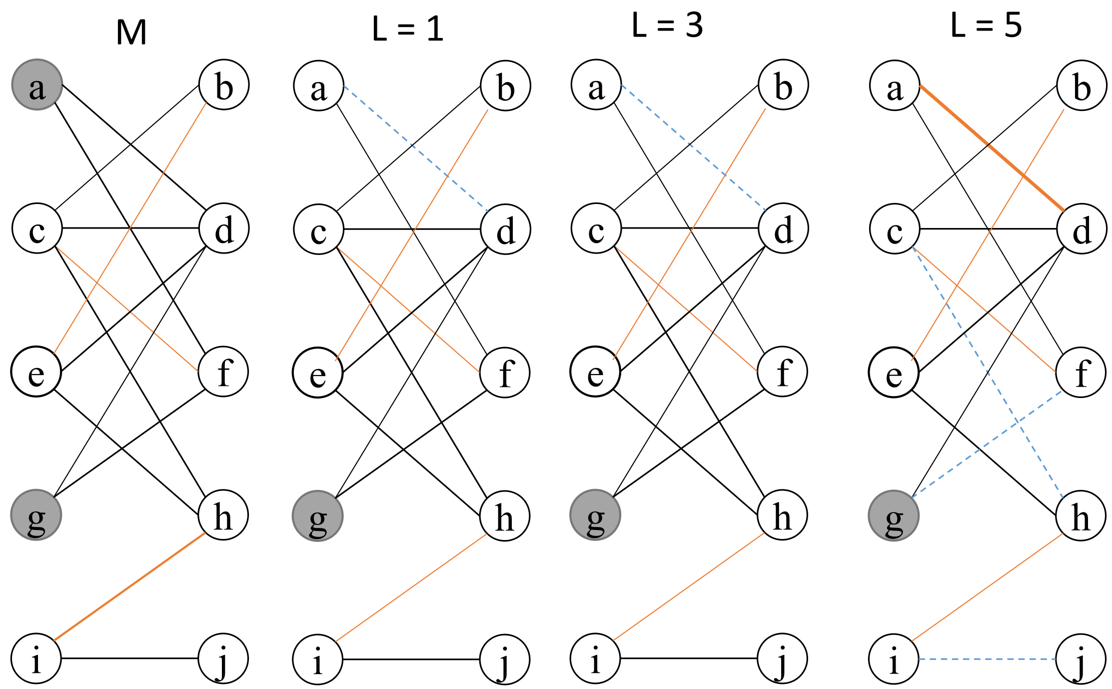




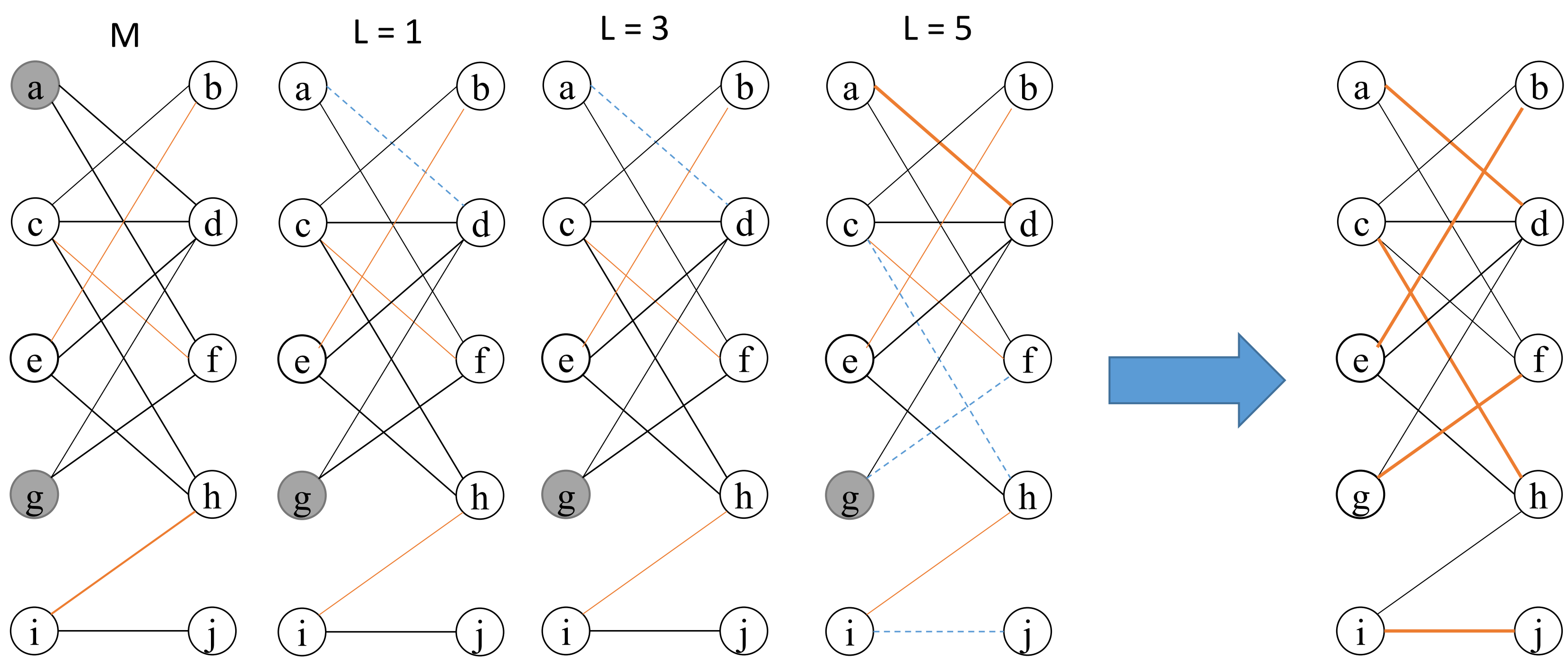
Example:



Example:

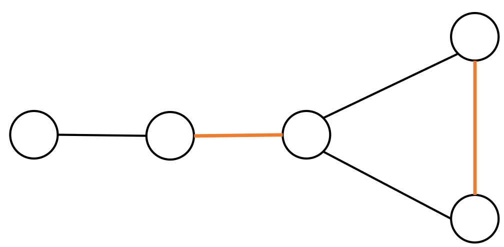


Example:



General Graphs

- What if we have odd cycles in the graph ?

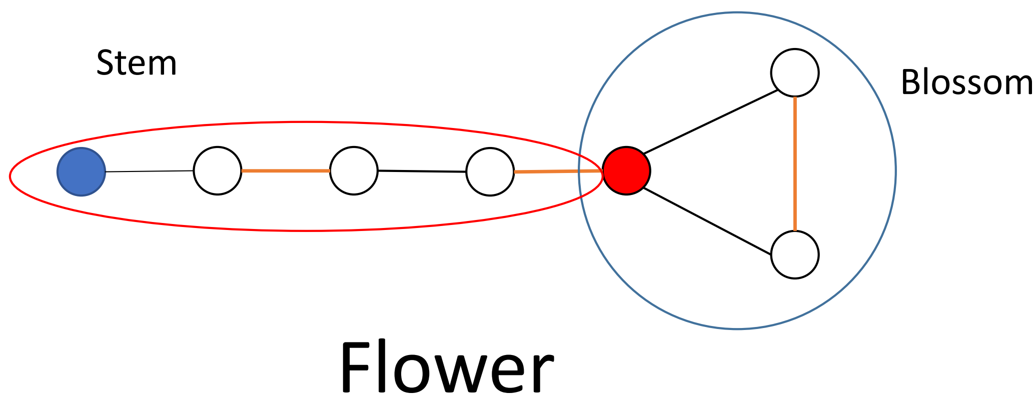


General Graphs

- Edmonds Algorithm
  - $O(n^2 m)$
- Gabow's Algorithm (1976), uses Edmonds Algorithm
  - $O(n^3)$
- Micali & Vazirani (1980)
  - $O(\sqrt{n}m)$
- Mucha & Sankowski
  - Randomization algorithm based on matrix multiplication
  - $O(n^{2.3})$

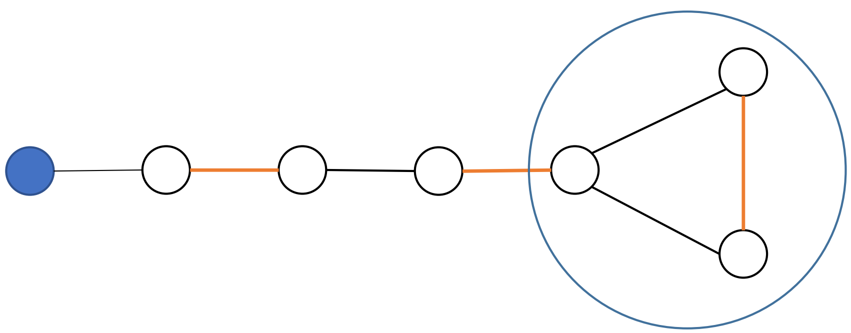
Blossoms

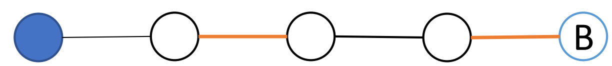
- It is an odd cycle with two adjacent edges to the stem and not in M with a unique exposed vertex (base).
- Stem is an even alternating path from an exposed vertex



Edmonds' Lemma

- Let G' and M' be obtained by contracting a blossom B in (G, M) to a single vertex.
- The matching M of G is maximum iff M' is maximum in G'.





Edmonds' Algorithm

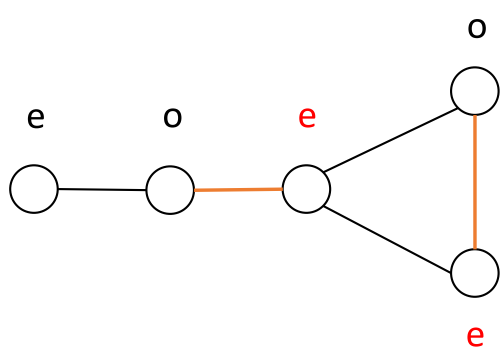
- If there exists an augmenting path corresponding to M or FLOWER :
  - If a Flower found:
    - Shrink the blossoms
  - Look for augmenting paths
  - If augmented path P found:
    - Add it to the matching M using symmetric difference $M' = M \oplus \mathrm{P}$
    - Then let $M = M'$

  If no augmenting paths found then we have a maximum matching in the bipartite graph
- Runs in (n^2m)

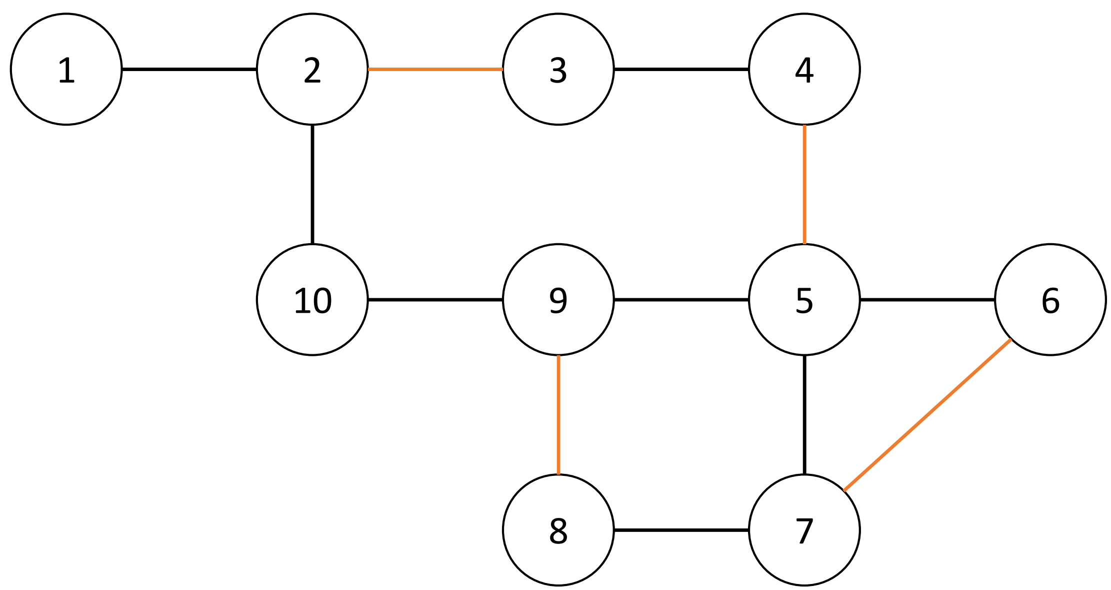
Detecting a blossom

- We do a traversing for alternating path just like the bipartite graph
  - Mark exposed vertex and at the even distance from it as (e)
  - Mark vertices at odd distances as (o)

- We have a blossom if we have two even vertices adjacent.

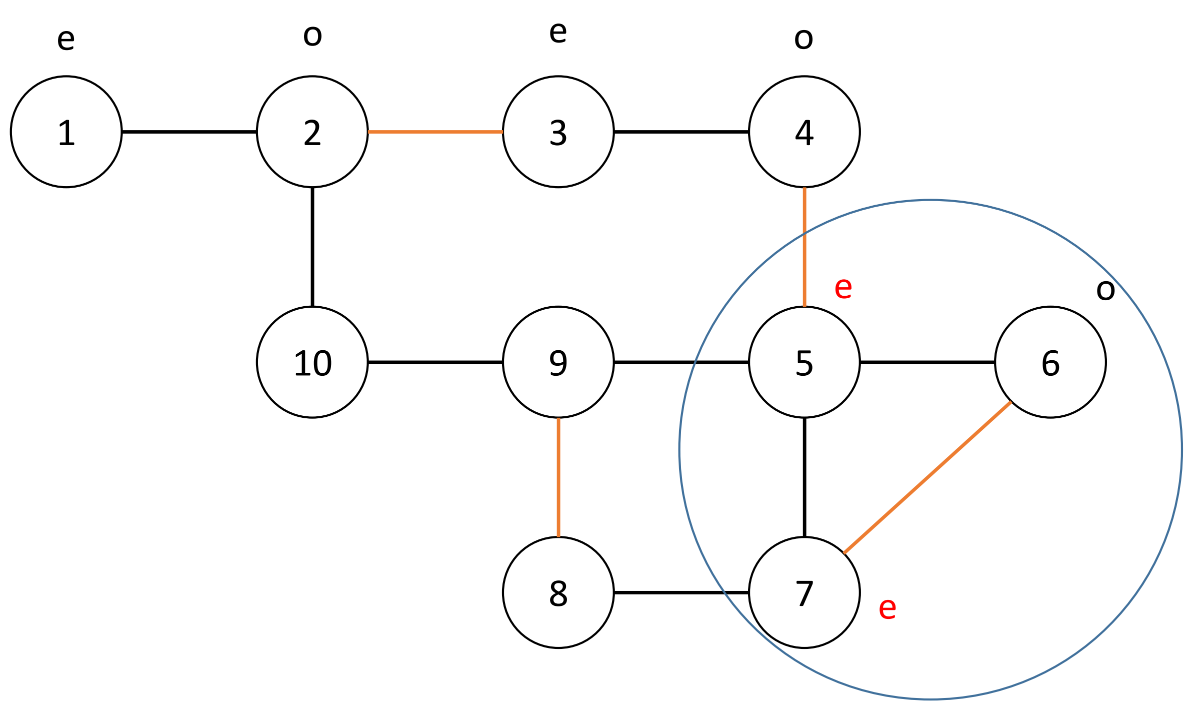


Example



$|M| = 4$

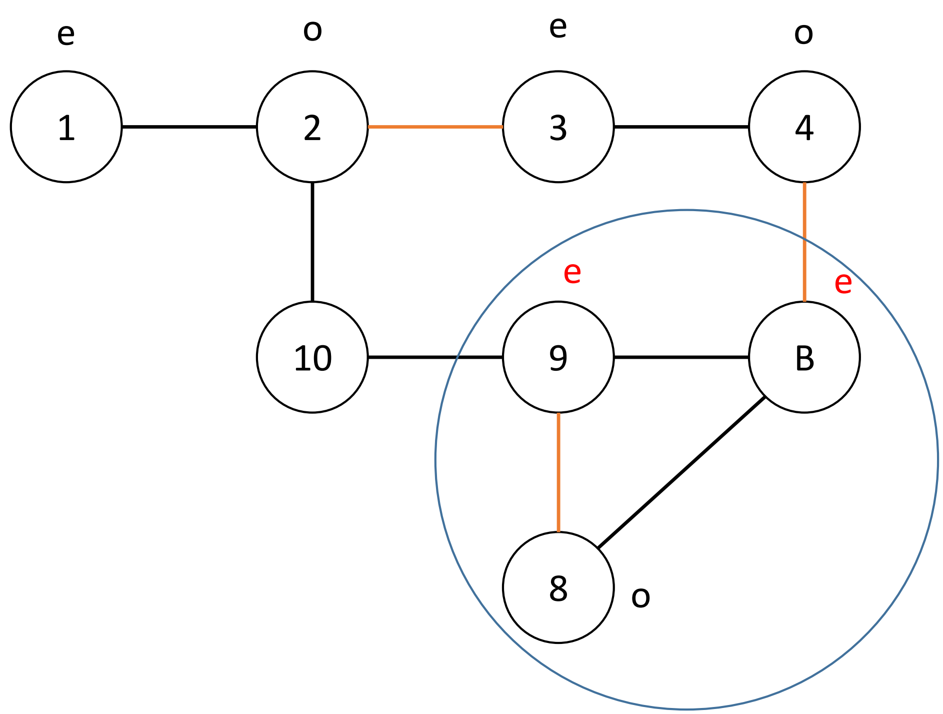
Example



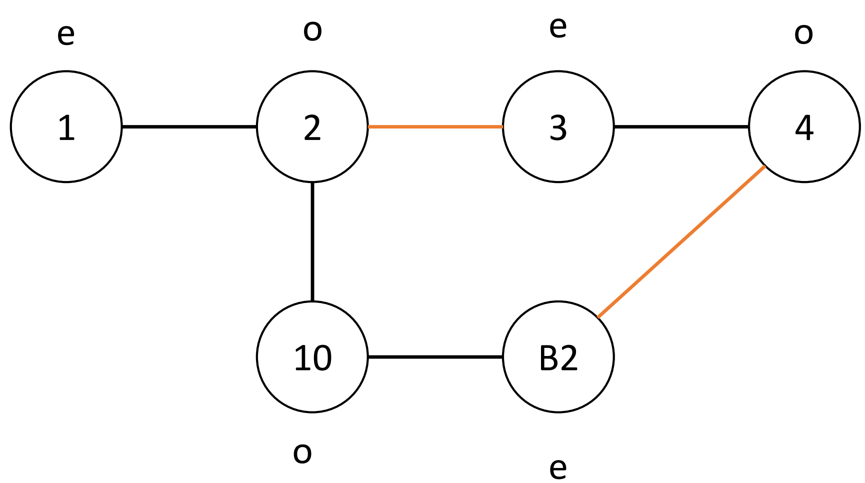
$$|M| = 4$$

Example

$|M| = 4$



Example



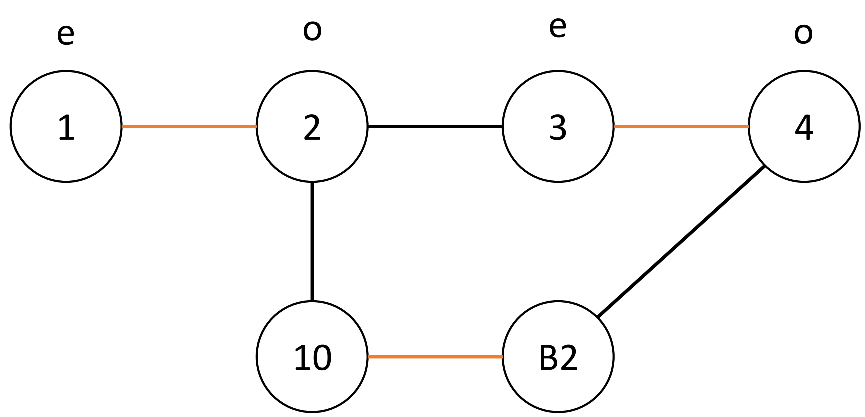
$$|M| = 4$$

$$B1 = 5, 6, 7$$

$$B2 = B1, 8, 9$$

Here we have an augmenting path after we compressed the blossoms to vertices

Example

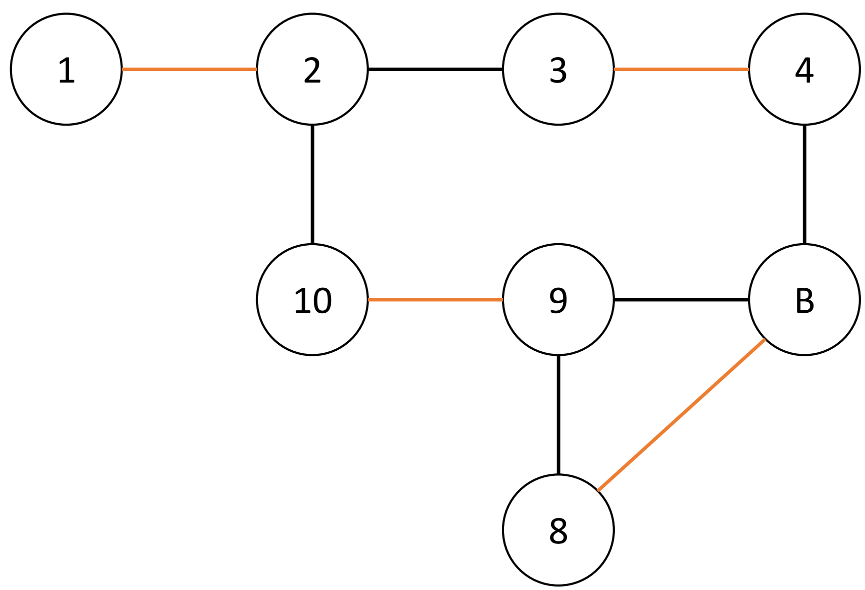


$$|M| = 4$$

$$B1 = 5, 6, 7$$

$$B2 = B1, 8, 9$$

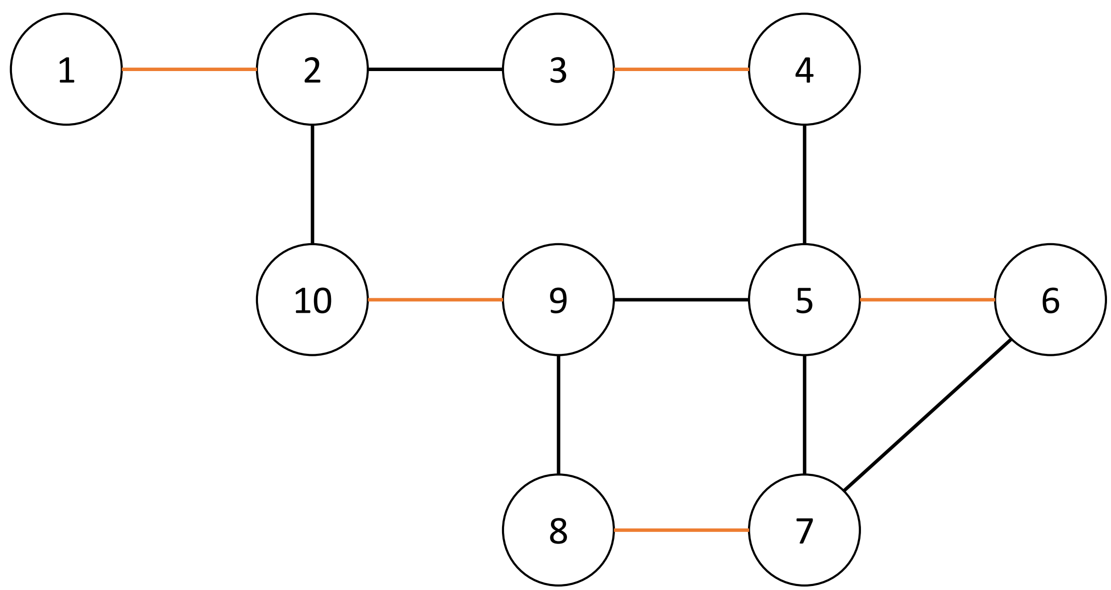
Example



$$|M| = 4$$

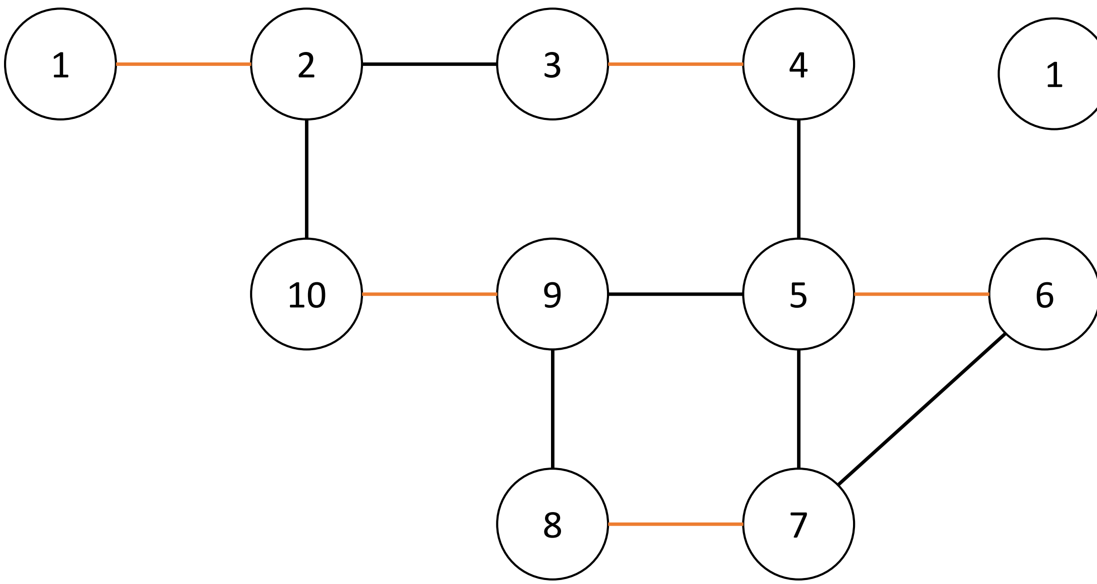
$$B1 = 5, 6, 7$$

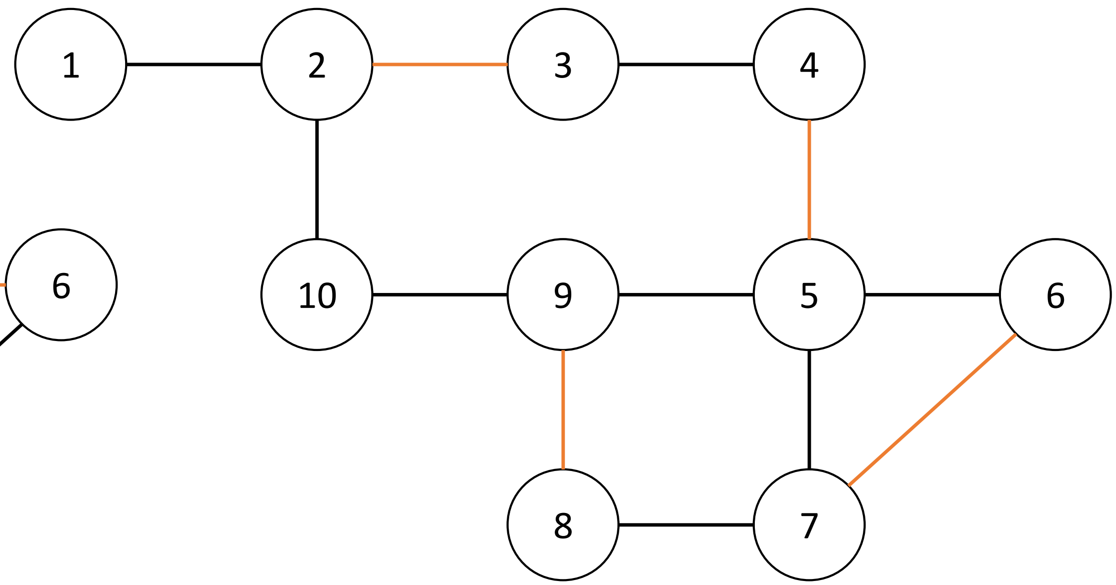
Example



$$|M| = 5$$

Example





REFERENCES

- Edmonds, Jack (1965), "Paths, Trees and Flowers", Canadian J. Math, 17: 449–467, doi:10.4153/CJM-1965-045-4, MR 0177907
- Norbert Blum, “A New Approach to Maximum Matching in General Graphs”.
- Galil Z.: Efficient Algorithms for Finding Maximum Matching in Graphs, Computing Surveys, 1986, 23--38.