<div style="border:1px solid black; padding:10px; text-align:center">

Lecture Notes: Maximum Matching in Bipartite Graphs

</div>

Objectives

- Be familiar with maximum matching problems and applications.

- Be able to solve maximum matching problem using Alternating Path Algorithm

Key Ideas

- Bipartite graphs

- Matching edges

- Maximum matching

- Perfect matching

- Cardinality of maximum matching

- Alternating paths

- Augmenting paths

1 Definitions

- A *bipartite graph* is a graph that can be partitioned (divided) into two sets (L and R) such that no two nodes in the same set are connected.

- A *matching edge* is an edge connecting one node from L to another node in R and a node cannot be "matched" by more than one node.

- A *matching* in a bipartite graph is a set of matching edges.

- A *maximum matching* is a matching with maximum cardinality (size).

- An *alternating path* is a path along which matching edges and non-matching edges alternate.

- An *augmenting path* is an alternating path starting with an exposed node in one side of a bipartite graph and ending at another exposed node in the other side.

2 Matching in Bipartite Graphs

Alternating Path Algorithm

1. Pick an *exposed* node (a node that has no matching edges adjacent to it) from either set of nodes (left or right). If there is none, we are done and the current matching is maximum.

2. Build an alternating tree starting from this node.

3. If another exposed node is reached, improve the matching by exchanging the matching and non-matching edges along the alternating path. This path is now called *augmenting path* between the two exposed nodes. Start over with the improved matching.

4. If no exposed node is reached, then repeat the above steps with another exposed node, but don't consider the nodes that are in the already built alternating trees (becasue we already know we cannot reach any exposed nodes from them)

Maximum Matching

P 733 example
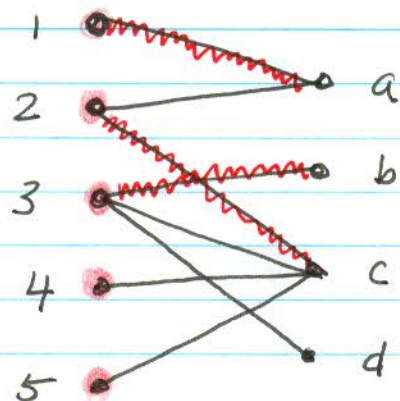
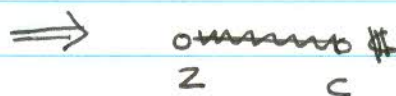1 Start with the first exposed node '1' and build alternating path.

exposed   exposed
   o ————— o
   1   a
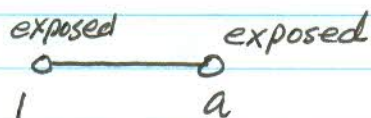
In case, it takes one edge to reach another exposed node 'a'

Convert into augmenting path

1   a

2 Start the second exposed node '2'
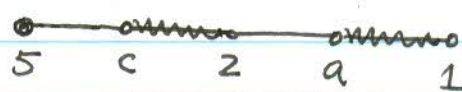
 $\Rightarrow$

2   c   2   c

3. Start with '3'

 $\Rightarrow$

3   b   3   b
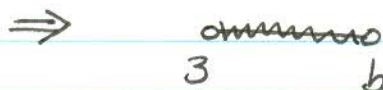
4 Start w/ node '4'

    no exposed reached

4   c   2   a   1
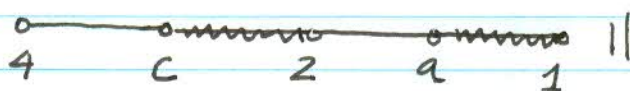
5 Start w/ node '5'.

    no exposed reached

5   c   2   a   1

cannot
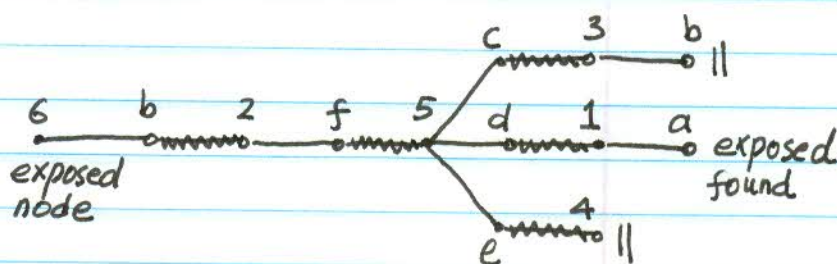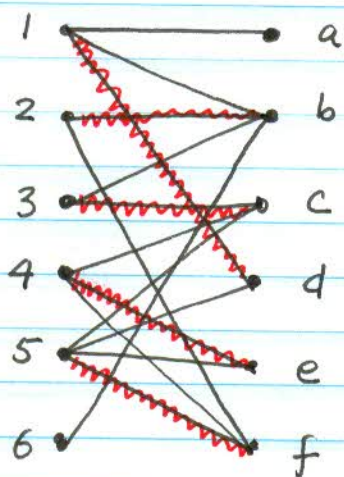
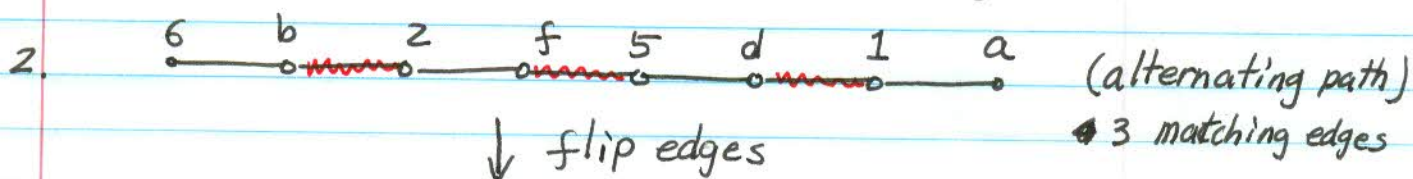This part has been tried in step 4

No more exposed node to start with. Done
Three matching edges are found.
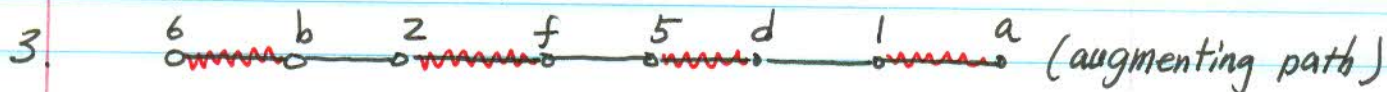
Bipartite graphs — max matching
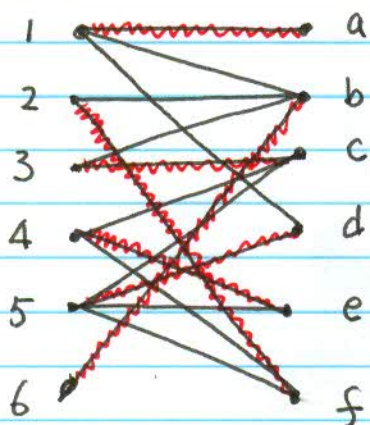Alternating path Algorithm

1. Suppose 5 matching edges marked in red have been found so for. There is one exposed node '6'. Start from node 6, build alternating path tree using BFS

2.   6   b   2   f   5   d   1   a   (alternating path)
                                                    ← 3 matching edges
        ↓ flip edges

3.   6   b   2   f   5   d   1   a   (augmenting path)

                                            now we have 4 matching edges

4. update the graph with newly found matching edges

Now no exposed nodes are left. Done!
We have 6 matching edges marked in red.