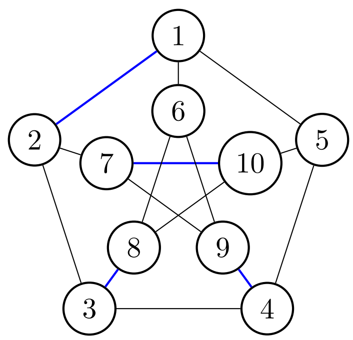
CS 6200: Algorithmics II	**Fall 2020**
Instructor: Avah Banerjee, S&T.	Scribe: Self
Lectures: 15	**Dates: 28 September**

1 Unweighted Matching in General Graphs

1.1 Matching Definitions

A *matching* denotes a function on edges in a graph $E(G)$ such that one edge must be match two vertices uniquely. In other words, a vertex may not be doubly matched, but it may be matched to no edges within a matching.

Fig. 1: The Petersen Graph, here with matching M as shown by the blue edges.



Definition 1.1 ***Free*** *vertices or edges are those which are not in a matching.*

Example 1.1 *Figure 1, contains a matching M. Vertices $\{1, 2, 3, 4, 7, 8, 9, 10\}$ are matched vertices, whereas vertices $\{5, 6\}$ are free under matching M.*

Definition 1.2 *An* ***alternating path*** *denotes a path in a graph taken such that a matched edge immediately follows an unmatched edge, and vice versa.*
Though a matching may match only an even number of vertices, alternating paths may be even or odd in length.

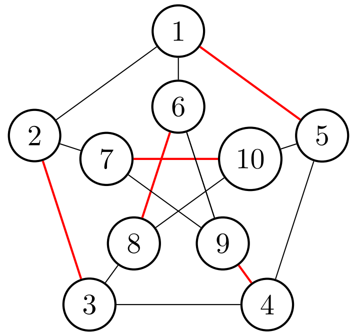
Example 1.2 *In Figure 1, an alternating path would be* $\{(1, 2), (2, 7), (7, 10), (10, 8), (8, 3), (3, 4), (4, 9)\}$

Definition 1.3 ***Augmenting paths*** *are alternating paths that both start and end with free vertices.*

Example 1.3 *In Figure 1, an augmenting path would be* $\{(5, 1), (1, 2), (2, 3), (3, 8), (8, 6)\}$

An augmenting path is significant, because the number of free edges is 1 more than the number of matched edges along the path. If a new matching could therefore be produced of greater size by including only the free edges of the prior matching.

Fig. 2: The Petersen Graph, here with an alternate matching M' as shown by the red edges.



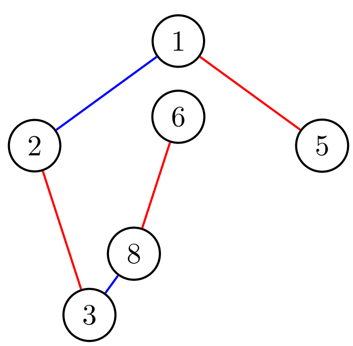
1.2 Symmetric Difference

Definition 1.4 *The* ***symmetric difference*** *obtained from two graphs A and B with $V(A) = V(B)$, denoted $A \oplus B$, is defined as the set of edges in A and the set of edges in B, but not in both. [1]*
The symmetric difference can be thought of as taking the exclusive or (XOR) of the edges in graphs with the same vertex set, or in this case, matchings.

Theorem 1.4 *If M is a matching and M' is its augmenting path on G, then $\Delta(M \oplus M') \leq 2$*

Proof: *Let M be a matching containing an alternating path, and M' be a matching containing an augmenting path relative to M. By the definition of a matching, each edge in the matching must match distinct vertices. So, in the edge-induced subgraph of M on G, each vertex would have degree 1. Suppose there is some vertex $v \in V(M \oplus M')$ such that $d(v) \geq 3$. This would mean that among M and M', there were 3 edges unique to either one of them that had v as an endpoint. But this would imply that v is doubly-matched in either M or M'! By contradiction, $\Delta(M \oplus M') \leq 2$.* ■

Fig. 3: The induced subgraph of $M \oplus M'$, with blue and red edges corresponding to M and M' respectively.



Theorem 1.5 *If a matching M is not maximum, then there must be an augmenting path.*
Proof: *Suppose M^* is a maximum matching. $|M| < |M^*|$. Let graph G' be the graph induced by*

$M \oplus M^*$. G' has one of these properties:
CASE I: *There is an even-length cycle. If an even-length cycle-exists, then within the cycle, the edges contained in it from M and M^* would be equal, since no two edges from the same set may be incident to the same vertex.*
CASE II: *There is an even-length path. Similarly to case 1, in an even-length path, the number of edges in M and M^* would be equal.*
CASE III: *There is an odd length path.*
Note that there is no case for an odd-length cycle; $M \oplus M^*$ could contain no such cycle, because G' contains only edges in $M \cup M^*$. Occurrence of an odd-length cycle would imply that at least $\lceil k/2 \rceil$ edges are contained in one of the matchings, where k represents the number of edges in the cycle. Since k is odd, one of the matchings is strictly greater in size than the other in the cycle. This guarantees two adjacent edges of the same matching, which violates the definition of a matching.

Suppose G' has an odd-length path. Since $|M| < |M^*|$, there exists some odd length path such that the starting and ending vertices are free in M.
This is the definition of an augmenting path for M! ■

1.3 Blossoms and Edmonds' Algorithm

We set out to find an algorithm that will find the maximum matching for a generalized graph G, through the use of augmenting paths. But we run into trouble, due to the existence of odd-length cycles. Odd-length cycles give differing path lengths when attempting to find an augmenting path, so we use an operation called ***shrinkage*** *in order to circumvent this issue.*

Definition 1.5 *Suppose $v, u \in V(G)$, adjacent to their neighbor sets $N(v)$ and $N(u)$.* ***Shrinking*** *v and u will form a single vertex $w \in V(G')$, whose neighbor set is $N(v) \cup N(U)$.*
Parity becomes important when attempting to find odd-length paths. We denote all free vertices as even, *and then in forming an alternating path, all vertices encountered from an even vertex are* odd, *and vice-versa. But this is too simplistic; an odd-length cycle will cause a vertex to be either of even or odd parity depending on the direction of labeling. This ambiguity is resolved through maintaining a data structure:*

1. *Maintain a collection of rooted trees.*
2. *Each vertex is either even, odd, or unreached.*
3. *The roots of the trees are free vertices. This is because if an augmenting path is to be found, it will begin with a free vertex.*
4. *All free vertices are even, all matched vertices are unreached.*
This gives us an initial condition. We devise the notion of a ***Blossom,*** *whereby an odd-length cycle contained in G is* shrunken *down to a single vertex.*

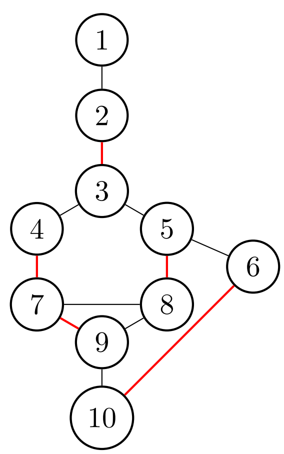
Tarjan provides the following algorithm: [1]

Example 1.6 *Suppose we start with the graph, with matched edges shown in red, shown below in figure 4.*

Algorithm 1 Blossom-Shrinking Algorithm

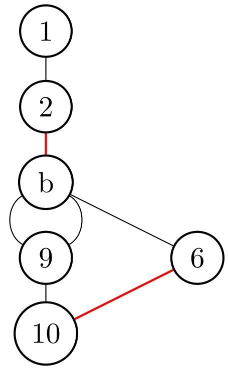
```
Input: V, E ∈ G(V, E), matching M.
Output: Augmenting path of M, M∗.
while Not found M∗, and some e ∈ E(G) are unexamined do
    Choose an unexamined edge (u, v) ∈ E(G) with v an even vertex.
    CASE I: v is odd. No operation.
    CASE II: v is unreached, and a matched vertex. Label it odd, and its neighbor set N(v) even.
    CASE III: v is even, u and v are members of different trees. Break, M∗ ← path through v from u to the root of the other tree.
    CASE IV: v is even, u and v are members of the same tree. (u, v) ∈ E(G) is a blossom.
    Let w be the nearest common ancestor of u and v. Shrink all vertices which are descendants of w and ancestors of u or v into a blossom b.
    Set the parity of the blossom b to w's parity.
end while
```

Fig. 4: An example graph for Blossom-Shrinking Algorithm.



The free vertices in this graph are $\{1, 9\}$*. Start with vertex* 1*. Examine edge* $(1, 2)$*. Since* 2 *is a matched vertex, label it odd, and label its matched vertex* 3 *as even. Now, move on to* 3*.*
Examine $(3, 4)$*,* $(3, 5)$ *in the same vein as edge* $(1, 2)$*; since* $(4, 7)$ *and* $(5, 8)$ *are matched, label* $\{4, 5\}$ *as odd and* $\{7, 8\}$ *as even. Now move on to* 7*.*
Examine $(7, 8)$*. This is an even node since it is free, and it is not a part of any other tree as of yet in our defined data structure. So this forms a blossom, b.*
The lowest common ancestor of $\{7, 8\}$ *is* 3*, and their mutual neighbor not in the descendants of* 3 *before them is* 9*. Let G' be the graph induced by shrinking* $\{3, 4, 5, 7, 8\}$ *into blossom b, shown below as figure 5.*

Fig. 5: G', after shrinkage onto blossom b.



Now, we can continue from vertex b.
Examine $(b, 9)$. *Since* 9 *is a free vertex, it is part of another tree, and since* b *is even and* 9 *is even and part of another tree, we reach case 3, and the path* $\{1, 2, b, 9\}$ *constitutes an augmenting path in* G'*, and the algorithm terminates.*

In the expanded blossom b*, such an augmenting path also exists in the path* $\{1, 2, 3, 5, 8, 9\}$. *This will always be the case if an augmenting path exists in the shrunken blossom graph, since one path through the expanded blossom will always contain more free edges than the other.*

References and Further Reading

[1] Tarjan, R. E., (1986). Data structures and network algorithms. Soc. for Industrial and Applied Mathematics. [Chapter 8, Chapter 9].