CS261: Exercise Set #3

For the week of April 13–17, 2015

Instructions:

(1) *Do not turn anything in.*

(2) The course staff is happy to discuss the solutions of these exercises with you in office hours or on Piazza.

(3) While these exercises are certainly not trivial, you should be able to complete them on your own (perhaps after consulting with the course staff or a friend for hints).

Exercise 11

Recall the main loop of Edmonds's blossom algorithm, which searches for an augmenting path P with respect to the current matching M. Formally prove the following invariants by induction on the number of inner while loop iterations (up to the point of an augmentation or contraction):

1. The predecessor pointers form a forest of directed in-trees, the roots of which are precisely the vertices not matched by M.
2. Every "even" or "odd" node belongs to exactly one of the in-trees; "unexplored" nodes belong to none of the trees.
3. Following the predecessor pointers from an even (respectively, odd) node back to its root yields an alternating path with an even (respectively, odd) number of edges.
4. Every odd node has in-degree 1 (i.e., is the predecessor of exactly one other node).
5. For every vertex v in an in-tree that is matched in M (to w, say), w belongs to the same in-tree, with either $pred[w] = v$ or $pred[v] = w$.
6. If (v, w) is an edge such that at least one of v, w has been explored (i.e., labeled either even or odd), and neither $pred[w] = v$ nor $pred[v] = w$, then $(v, w) \notin M$.

Now re-read the proof sketch of the correctness of Edmonds's algorithm for the bipartite case. Does it seem more convincing in light of these invariants?

Exercise 12

Consider the Search subroutine of Edmonds's algorithm in the special case of bipartite graphs (where no odd cycle contractions occur). Give an $O(m)$-time implementation of this subroutine (yielding an $O(mn)$-time algorithm for the maximum bipartite matching problem).

Exercise 13

Prove that, in bipartite graphs, one can bypass Edmonds's search procedure and instead compute an augmenting path (with respect to the current matching M) using standard graph search (e.g., BFS or DFS) in a suitable directed graph.

[Hint: Direct the edges of M in one direction and the other edges in the other direction.]

Exercise 14

Suppose you are given an undirected graph $G = (V, E)$ and a positive integer b_v for every vertex $v \in V$. A *b-matching* is a subset M of edges such that each vertex v is incident to at most b_v edges of M. (The standard matching problem corresponds to the case where $b_v = 1$ for every $v \in V$.)

Prove that the problem of computing a maximum-cardinality b-matching reduces to the problem of computing a (standard) maximum-cardinality matching in a bigger graph. Your reduction should run in time polynomial in the size of G and in $\sum_{v \in V} b_v$.

Exercise 15

Let $G = (V, E)$ be an undirected graph and M a (not necessarily maximum) matching in G. Let $S \subseteq V$ denote the set of $2|M|$ vertices that are matched in M. Prove that there exists a maximum-cardinality matching in which every vertex of S is matched.

[Hint: what can you say about the set of currently matched vertices throughout a trajectory of Edmonds's algorithm?]