Fundamental Algorithms
Courant, Fall 2017

Homework 9

Instructor: Amir Shpilka
Due Date: Wed Nov 15, 11:55pm

16.3-2 1. (3 points) Prove that a binary tree that is not full cannot correspond to an optimal prefix code.

2. (2 points) Construct an optimal Huffman code (by showing the tree) for the following set of alphabets and frequencies:

   `a:32 b:5 c:12 d:10 e:6 f:15 g:32 h:21`

3. (a) (4 points) Design $O(n)$ algorithm to test if a given *undirected* graph G on n nodes is acyclic. Notice, the running time of your algorithm should not depend on the number of edges m!

   (b) (3 points) Extend the above algorithm to actually print the cycle, in case G is cyclic.

4. You have an undirected graph $G = (V, E)$ and two special nodes $r, d \in V$. At time 0, node r is republican, node d is democratic, while all the other nodes $v \notin \{r, d\}$ are initially "undecided". For every $i = 1, 2, 3, \ldots$, the following 2-stage "conversion" process is performed at time time i. At the first stage, all republicans at time $(i - 1)$ look at all their neighboring nodes v which are still undecided, and convert those undecided nodes to become republican. Similarly, at the second stage, all democratic nodes at time $(i - 1)$ look at all their neighboring nodes v which are still undecided by the end of the first stage above, and convert those undecided nodes to become democratic. The process is repeated until no new conversions can be made. For example, if G is a 5-cycle $1, 2, 3, 4, 5$ where $r = 1$, $d = 5$, after time 1 node 2 becomes republican and node 4 becomes democratic, and after time 2 the last remaining node 3 becomes republican (as republicans move first). On the other hand, if the initial democratic node was $d = 3$ instead, then already after step 1 nodes 2 and 5 become republican, and node 4 becomes democratic, and no step 2 is needed.

   Assume each node v have a field $v.color$, where *red* means republican, *blue* means democratic, and *white* means undecided, so that, at time 0, $r.color = red$, $d.color = blue$, and all other nodes v have $v.color = white$.

   (a) (5 points) Using two BFS calls, show how to properly fill the final color of each node.

   (b) (8 points) Show how speed up your procedure in part (a) by a factor of 2 (or more, depending on your implementation) by directly modifying the BFS procedure given in the book. Namely, you must compute the final colors of each node by performing a *single*, appropriately modified, BFS traversal of G. Please write pseudocode similar to the standard BFS pseudocode, and briefly explain your code (no proof needed).

   (c) (5 points) Now assume that at time 0 more than one node could be republican or democratic. Namely, you are given as inputs some disjoint subsets R and D of V, where nodes in R are initially republican and nodes in D are initially democratic, but otherwise the conversion process is the same. For concreteness, assume $|R| = |D| = t$ for some $t \geq 1$ (so that parts (a) and (b) correspond to $t = 1$). Show how to generalize your solutions in parts (a) and (b) to this more general setting. Given parts (a) and (b) took time $O(|V| + |E|)$ (with different constants), how long would their modifications take as a function of t, $|V|$, $|E|$? Which procedure gives a faster solution?