Homework Assignment #6

Due Time/Date

2:10PM Tuesday, November 17, 2020. Late submission will be penalized by 20% for each working day overdue.

Note

Please write or type your answers on A4 (or similar size) paper. Drop your homework by the due time in Yih-Kuen Tsay's mail box on the first floor of Management College Building 2. You may discuss the problems with others, but copying answers is strictly forbidden.

Problems

There are five problems in this assignment, each accounting for 20 points. (Note: problems marked with "(X.XX)" are taken from [Manber 1989] with probable adaptation.)

1. (6.62) You are asked to design a schedule for a round-robin tennis tournament. There are $n = 2^k$ ($k \geq 1$) players. Each player must play every other player, and each player must paly one match per round for $n-1$ rounds. Denote the players by $P_1, P_2, \ldots, P_n$. Output the schedule for each player. (Hint: use divide and conquer in the following way. First, divide the players into two equal groups and let them play within the groups for the first $\frac{n}{2} - 1$ rounds. Then, design the games between the groups for the other $\frac{n}{2}$ rounds.)

2. Consider the solutions to the union-find problem discussed in class. Suppose we start with a collection of ten elements: A, B, C, D, E, F, G, H, I, and J.

   (a) Assuming the balancing, but not path compression, technique is used, draw a diagram showing the grouping of these ten elements after the following operations (in the order listed) are completed:

      i. union(A,B)
      ii. union(C,D)
      iii. union(A,D)
      iv. union(E,F)
      v. union(G,H)
      vi. union(F,G)
      vii. union(I,J)
      viii. union(H,I)
      ix. union(D,J)

In the case of combining two groups of the same size, please always point the second group to the first.

(b) Repeat the above, but with both balancing and path compression.

3. (6.21) The input is a set S with n real numbers. Design an $O(n)$ time algorithm to find a number that is *not* in the set. Prove that $\Omega(n)$ is a lower bound on the number of steps required to solve this problem.

4. (6.32) Prove that the sum of the heights of all nodes in a complete binary tree with n nodes is at most $n-1$. (A complete binary tree with n nodes is one that can be compactly represented by an array A of size n, where the root is stored in $A[1]$ and the left and the right children of $A[i]$, $1 \leq i \leq \lfloor \frac{n}{2} \rfloor$, are stored respectively in $A[2i]$ and $A[2i+1]$. Notice that, in Manber's book a complete binary tree is referred to as a balanced binary tree and a full binary tree as a complete binary tree. Manber's definitions seem to be less frequently used. Do not let the different names confuse you. "Balanced binary tree" in the original problem description is the same as "complete binary tree")

5. Consider the *next* table as in the KMP algorithm for string $B[1..9] = abaababaa$.

1	2	3	4	5	6	7	8	9
a	b	a	a	b	a	b	a	a
-1	0	0	1	1	2	3	2	3

Suppose that, during an execution of the KMP algorithm, $B[6]$ (which is an a) is being compared with a letter in A, say $A[i]$, which is not an a and so the matching fails. The algorithm will next try to compare $B[next[6]+1]$, i.e., $B[3]$ which is also an a, with $A[i]$. The matching is bound to fail for the same reason. This comparison could have been avoided, as we know from B itself that $B[6]$ equals $B[3]$ and, if $B[6]$ does not match $A[i]$, then $B[3]$ certainly will not, either. $B[5]$, $B[8]$, and $B[9]$ all have the same problem, but $B[7]$ does not.

Please adapt the computation of the *next* table so that such wasted comparisons can be avoided.