CS 5633 Analysis of Algorithms – Spring 10

3/25/10

7. Homework

Due **Thursday 4/1/10** (seriously!) before class

Justify all your answers.

1. **Greedy scheduling (6 points)**

   Suppose you are the manager of a train station. You are given a sorted set $A = a_1, \ldots, a_n$ of n points in time which are the times when trains will arrive at your station. Only when a train arrives you need to have employees at the station. Due to union rules, each employee can work at most one hour at the station. The problem is to find a scheduling of employees (i.e., a set of one-hour intervals) that covers all the times in A and uses the fewest number of employees.

   Prove or disprove that the two greedy approaches below correctly solve this problem. (Only one is correct, and the correctness can be proved using a copy-paste argument. In order to disprove give a counterexample.)

   (a) Let I be an interval that covers the most number of points in A. Add I to the solution and recursively continue on the points in A not covered by I.

   (b) Let a_j be the smallest point in A. Add the interval $I = (a_j, a_j + 1)$ to the solution, and recursively continue on the points in A not covered by I.

2. **Binary Counter (3 points)**
   Use aggregate analysis to show that, over a sequence of n increment operations on a binary counter, the amortized runtime of one such increment operation is $O(1)$.
   (Hint: Study the flipping behavior of every single bit $A[i]$.)

3. **Queue from Two Stacks – (7 points)**

   Assume we are given an implementation of a stack, in which PUSH and POP operations take constant time each. We now implement a queue using two stacks A and B as follows:

   ENQUEUE(x):
   • Push x onto stack A

   DEQUEUE():
   • If stack B is nonempty, return B.POP()
   • Otherwise pop all elements from A and while doing so push them onto B. Return B.POP()

   a) (2 points) Show how the following sequence of operations operates on the two stacks. Suppose the stacks are initially empty.
   Enqueue(1), Enqueue(2), Enqueue(3), Enqueue(4), Dequeue(), Dequeue(), Enqueue(5), Enqueue(6), Dequeue()

   b) (2 points) Why is the algorithm correct? Argue which invariants hold for A and B.

   c) (3 points) Prove using the accounting method that the amortized runtime of ENQUEUE and DEQUEUE each is $O(1)$. Argue why your account balance is always non-negative.

FLIP OVER TO BACK PAGE $\Longrightarrow$

4. **Union-Find (4 points)**

```
for(i=1; i<=15; i++) x[i]=MAKE-SET(i);
UNION(x[1],x[2]); UNION(x[4],x[5]); UNION(x[9],x[10]); UNION(x[12],x[13]);
UNION(x[1],x[3]); UNION(x[5],x[9]); UNION(x[11],x[12]); UNION(x[14],x[15]);
UNION(x[2],x[10]); UNION(x[11],x[15]);
UNION(x[10],x[13]);
FIND-SET(x[14]);
```

   Assume an implementation of the Union-Find data structure with a disjoint-set forest with union-by-weight and path compression.

   Show the data structure after every line of code. What is the answer to the `FIND-SET` operation?

5. **Ackermann (1 points)**
   What is the value of $\alpha(10^{10000})$? Justify your answer.

6. **BFS and DFS with Adjacency Matrix (4 points)**
   Suppose the graph $G = (V, E)$ is given in an adjacency matrix. How fast do BFS and DFS run on this graph? Justify your answer.