EECS 3101A Fall 2024-25 – Test 2

Instructor: Jeff Edmonds

80 min/marks. Keep answers short and watch your time.

1. Recursion

   (a) (11 min/marks) Complete this recursive code.

   algorithm $Smallest(tree, k)$

   $\langle$***pre−cond***$\rangle$**:** $tree$ is a binary tree and $k \geq 1$

   $\langle$***post−cond***$\rangle$**:** Outputs $\langle kth, ??\rangle$ where kth is the k^{th} value if the values in the nodes were printed out in Infix Binary Search Tree order (or $-\infty$ otherwise).
   It can output something else as well if you like.

   begin

   - Answer: We also return the number of nodes in the tree.

```
    if($tree = emptyTree$) then
        return $\langle -\infty, 0\rangle$
    elseif
        $\langle kth_{left}, n_{left}\rangle = Smallest(leftSub(tree), k)$
        $k = k - kth_{left}$
        if ($k == 1$) then $kth_{root} = tree.value$ else $kth_{root} = -\infty$
        $k = k - 1$
        $\langle kth_{right}, n_{right}\rangle = Smallest(rightSub(tree), k)$
        $n = n_{left} + 1 + n_{right}$
        $kth = max(kth_{left}, kth_{root}, kth_{right})$
        return $\langle kth, n\rangle$
    endif
end alg
```

   (b) (5 min/marks) Give about 5 sentences arguing that your program works.

   - Answer: Note how we add up the number of nodes and adjust the value k by the number of nodes as we move past the left subtree and the root. We give both friends smaller instances that meet the precondition and hence we can trust their answers without **micromanaging** them. The left, the root, and the right will give us $-\infty$ unless they are the one with the value. The max will pick out that value. If the input is too small, i.e., empty tree, we give the right answer.

2. (4×12=48 min/marks) Quick Answers. For each, your answer need only be a few sentences.

Deleted What are Jeff's key mantras for understanding recursive algorithms?

   - Answer: Give any instance to friends as long as they are smaller and meet the pre conditions.
     Trust your friends to give the correct answers for their instances.
     Do not micromanage your friends.
     If your input is so small you cant make it smaller, then solve it yourself.

   (a) Suppose the running time of a program is $T(n) = 9T(n/3) + n^2$.
   What in the program is 9?
   What in the program is 3?
   Solve $T(n)$.

   - Answer: The number of recursive calls (friends) is 9.
     The size of the subinstances given to these calls is $\frac{1}{3}$ the size of the given instances.
     $\frac{\log_3 9}{\log_3 3} = 2 = c$. Hence the work done at the top is the same as the number of basecases. But there are $\log_3 n$ levels of recursion. Hence $T(n) = \Theta(n^2 logn)$.

   (b) Name an algorithm taught in class whose running time is given by $T(n) = 2T(n/2) + \Theta(n)$. What is $\Theta(T(n))$? Don't show work.

- Answer: Merge/Quick Sort. $\Theta(n \log n)$.

(c) Name an algorithm taught in class whose running time is given by $T(n) = 3T(n/2) + \Theta(n)$. No calculator. Approximate $\Theta(T(n))$. Don't show work.

- Answer: The first algorithm for multiplying that beat $\Theta(n^2)$. $log(a)/log(b) = log(3)/log(2) \approx 1.58$ giving time $\Theta(n^{1.58})$.

(d) Name an algorithm whose running time is given by $T(n) = 2T(n-1) + \Theta(1)$. What is $\Theta(T(n))$? Don't show work.

- Answer: Towers of Hanoi. $\Theta(2^n)$.

Deleted Name an algorithm taught in class whose running time is given by $T(n) = T(n-1) + \Theta(n)$. What is $\Theta(T(n))$? Don't show work.

- Answer: Insertion Sort. $\Theta(n^2)$.

(e) What is the time complexity of the following algorithm? Why?

algorithm $R(N)$
⟨***pre−cond***⟩**:** Integer Value
⟨***post−cond***⟩**:** Say Hi

```
begin
    loop $i = 1$ to $N$
        loop $j = 1$ to $N$
            Print("Hi")
        end loop
    end loop
end algorithm
```

- Answer: The size of the input is measured in bits, ie, $n = log(N)$.
  The value of the input $N = 2^n$ is exponential in this size.
  The time as a function of input size is $T(n) = N^2 = (2^n)^2 = 2^{2n}$.

(f) What are the general loop invariants for searching from the single source node s to every other node? What was the name Jeff gave to the proof that all nodes reachable from s are found.

- Answer: Each Found node is reachable from s. Each handled node has all of its neighbors found. The proof was called the Great Wall of China.

(g) What is the extra LI for Depth First Search?
How is it maintained?

- Answer: What is on the stack forms a path from s to the current node. (Nodes popped off the stack have been completely handled and nodes not yet on the stack not handled at all.) This is maintained as follows: If the current node has an unhandled child, then it is pushed on extending the path. If not, then the current node is popped shrinking the path.

(h) Given a directed graph with edge weights and special nodes s and t, give algorithm for min cut. How do you prove to your boss that there is a solution as good as you have found and not a better one.

- Answer: Run the max flow algorithm. It returns a flow and a cut of matching value.
  The cut you found witnesses that such a good cut can be obtained.
  Because flow you found has a matching value, it witnesses that no cut can be better.

(i) Can a recursive back tracking algorithm take exponential time? How?

- Answer: A recursive back tracking algorithm can take exponential time because it effectively tries all possible solutions of which there are an exponential number.

(j) Can a hill climbing algorithm take exponential time? How?

- Answer: A hill climbing algorithm can take exponential time if the value of the current solution gets better by only one each iteration and this value in an $\mathcal{O}(n)$ bit number.

(k) What is the input and output of the room scheduling greedy algorithm that we covered? What is the greedy algorithm for room scheduling that works? What is the motivation? (No proof.)

- Answer: The input is a set of events each of which has a start time and a finishing time and the goal is to schedule as many non-overlapping events into a single room as possible.
  The working greedy algorithm is Earliest Finishing Time: Schedule the event which will free up your room for someone else as soon as possible.

(l) Name reasonable two greedy algorithms that do not work for the room scheduling problem along with a proof for each.

- Answer: Shortest event counter example: A short event overlaps with the end of one long event and the beginning of the next long event. This greedy algorithm can only take the short one, but the optimal takes the two long ones.
  Earliest Starting Time counter example: The earliest one stays for a long time so no other event can be scheduled.
  Conflicting with the Fewest Other Events counter example: The events in the optimal solution have lots of copies of the same events, to make them look bad.

3. (16 min/marks) Give the $\forall/\exists$ logic statement that is the generic "maintain the loop invariant" statement with the generic loop invariant for a greedy algorithm inserted. Play the prover, adversary, oracle gave to prove this statement under the following scenario. The mom wants the son to do well in life but he fails his law test today. A friend must convince her that the son has not gone wrong. Be sure to identify what every variable in the statement represents in the story.

- Answer: The generic "maintain the loop invariant" statement is $[LI_{t-1}\&\neg Exit\&code] \rightarrow LI_t$.
  The generic loop invariant for a greedy algorithm is "We have not gone wrong. (If there is a solution), then there is at least one optimal solution S_t consistent with the choices A_t made so far by the algorithm."
  We combine these to give
  [$\exists$ solution S_{t-1} that is
  - consistent with algorithm's choices A_{t-1}
  - valid
  - (optimal)

  There is another decision to make.
  Code makes a decision at time t, extending A_{t-1} to A_t]
  $\rightarrow$ [$\exists$ solution S_t that is
  - consistent with algorithm's choices A_t
  - valid
  - $val(S_t) \geq val(S_{t-1})$ (optimal)]

  Here A_{t-1} is the description of everything the son did before now and S_{t-1} is his mom dreams of him being a lawyer for his life.
  We assume the LHS of this statement. The oracle assures us of it.
  The prover needs to prove the RHS.
  The prover proves $\exists S_t$ by constructing it as follows:
  - The oracle gives him the S_{t-1} that she assures exists.
  - The oracle tells him that the action of the algorithm at time t is that the son failed the lawyer exam.
  - The prover massages S_{t-1} into S_t. He replaces the fact in S_{t-1} that the mom expected her son to pass this exam to be consistent with the reality that he failed it. He also replaces the mom's dream that her son's future is to be a lawyer to be her new dream that he is to be a doctor.

The prover proves S_t is consistent with A_t as follows:

- The oracle assures him that S_{t-1} was consistent with A_{t-1}.
- He made sure when building S_t that it does not mess up being consistent with A_{t-1}.
- He made sure S_t is consistent with the algorithm's decision at time t.

The prover proves S_t is valid as follows:

- The oracle assures him that S_{t-1} was valid.
- He made sure when building S_t that it does not mess this up. Being a doctor instead is still valid.
- The prover proves $val(S_t) \geq val(S_{t-1})$ because in the mom's eyes being a doctor is as good as being a lawyer.

This completes the game.