CS 535 Design and Analysis of Algorithms **Fall Semester, 2025**

Homework 2

Assigned: September 9 *Due: September 23*

All students: please use Canvas to submit your solutions.
Notes for pseudocode usage:

1. C/Java instructions are fine. But do not write object-oriented additions. Do not declare or use any class. Declare only procedures (if necessary) and explain in words what each procedure does, and what is the use of each parameter. Feel free to use as procedures algorithms from the textbook; indicate page and edition. Please avoid Python as the running time can be obscured.

2. One instruction per line

3. Match the brackets with a horizontal line

4. Number your lines

5. Write down if your array is indexed $0 \ldots n-1$ or $1 \ldots n$.

Problem 1 This problem refers to binary search trees as defined in the Chapter 12 of the textbook.

Show that one TREE-MINIMUM followed by $(n-1)$ calls to TREE-SUCCESSOR takes $\Theta(n)$ times. In which order are the nodes visited?

Problem 2 This problem refers to binary max-heaps as defined in the Chapter 6 of the textbook, and the operations described and implemented there.

We want to claim an amortized cost of $O(1)$ for EXTRACT-MAX, which is actually possible with an amortized cost of $O(\lg n)$ for INSERT. Find a potential function Φ to obtain these bounds.

Problem 3 We describe below a data structure that maintains the transitive closure of a directed graph while arcs (directed edges) are added to the graph.

Formally, a set of vertices V is given (with $|V| = n$), and arcs $e_1, e_2, \ldots, e_m$ become available one by one (e_i is not known before computing R_{i-1}, defined below). Let $G_i = (V, E_i)$, where $E_0 = \emptyset$ and $E_i = E_{i-1} \cup e_i$. Let R_i, a $n \times n$ matrix, have $R_i[u, v] = 1$ if u has a directed path to v, and $R_i[u, v] = 0$ otherwise. Thus R_i stores the transitive closure of G_i.

Note that R_0 has entries that are 1 only on the main diagonal.

1. Give a series of instances (one for each n) such that there exists an i with the number of entries 1's in R_i being $\Omega(n^2)$ higher than the number of entries 1 in R_{i-1}.

2. Consider however the code

   ADD(e_i) where the tail of e_i is u and the head of e_i is v:

   ```
   for all x ∈ V
     if R[x,u] = 1 AND R[x,v] = 0
       for all y ∈ V
         R[x,y] ← max(R[x,y], R[v,y])
   ```

   Prove that if $R = R_{i-1}$ before the code is executed, then $R = R_i$ after the code is executed.

3. Use the first part of this problem to show that ADD(e_i) may have running time $\Omega(n^2)$.

4. Prove that despite this, the running time of m operations ADD(.) is $O(nm + n^3)$.