Ontario Tech Science

Kourosh Davoudi
kourosh@ontariotechu.ca

Midterm

CSCI 3070U: Design and Analysis of Algorithms

Instructions

- The exam is online and will take place on campus (for the location see the canvas).
- You can have one A4 handwritten cheat sheet for the exam (no other materials are allowed).
- Any sign of academic misconduct will be followed up and can have a serious academic penalty. It is the responsibility of students to be aware of the actions that constitute academic misconduct. Please see:
- http://calendar.uoit.ca/content.php?catoid=22&navoid=879#Academic_conduct
- You have 80 minutes to earn 30 points.
- **The due time is 3:30 PM** (except you have an accommodation letter), and **you should submit it before due time**. However, the exam is **available till 4:00 PM** for very special circumstances like power outages, computer crashes etc.
- Please note that you have only one attempt.
- If there is a technical problem and you submit answers after the due time, explain the situation in the last text box after all questions. You should provide reasonable evidence.
- You need to write the answers to each question in the provided space (**please type it**). Just typing the answers is enough.
- Do not spend too much time on any problem.
- Pay close attention to the instructions for each problem and just answer what is requested.
- Good Luck!

Learning Outcomes

- Understanding the order of growth
- Solving recurrence equations
- Finding the time complexity of the algorithm
- Tracing the algorithms and understanding their procedure
- Designing an algorithm

Understanding the order of growth

$$f_1(n) = 8\sqrt{n}, \quad f_2(n) = 25^{1000}, \quad f_3(n) = (\sqrt{3})^{\lg n}$$

Solution: f_2, f_1, f_3

Solving recurrence equations

$$T(n) = 9\,T(n/3) + \Theta\,(n \log n)$$

Solution: Because $n^{\log_3 9} = n^2$, and n^2 is asymptotically greater than n log n by more than polylogarithmic factor, this is case 1 of the Master Theorem, and $T(n) = \Theta(n^2)$.

For the example of recursion tree see the video on canvas.

Finding the time complexity of the algorithm

$\textsc{Ternary-Search}(x, A, i, j)$

// *Assumption:* $A[i] \leq x < A[j]$
if $j - i \leq 1$:
  return i
$p = \frac{2}{3}i + \frac{1}{3}j$
$q = \frac{1}{3}i + \frac{2}{3}j$
if $x < A[p]$:
  return $\textsc{Ternary-Search}(x, A, i, p)$
elseif $A[p] \leq x < A[q]$:
  return $\textsc{Ternary-Search}(x, A, p, q)$
elseif $x \geq A[q]$:
  return $\textsc{Ternary-Search}(x, A, q, j)$

$$\boxed{\begin{aligned} T(n) &= T(n/3) + 1 \\ T(1) &= 1 \end{aligned}}$$

Designing an algorithm

- Design an algorithm that takes a sorted array $\mathbf{A}$ and returns the location of an element such that $\mathrm{A}[i] = i$ (Otherwise null)