CS3230 Tutorial 7

Midterm Review

(AY 25/26 Semester 2)

March 9, 2026

(Prepared by Benson, some questions stolen from Xing Chen)

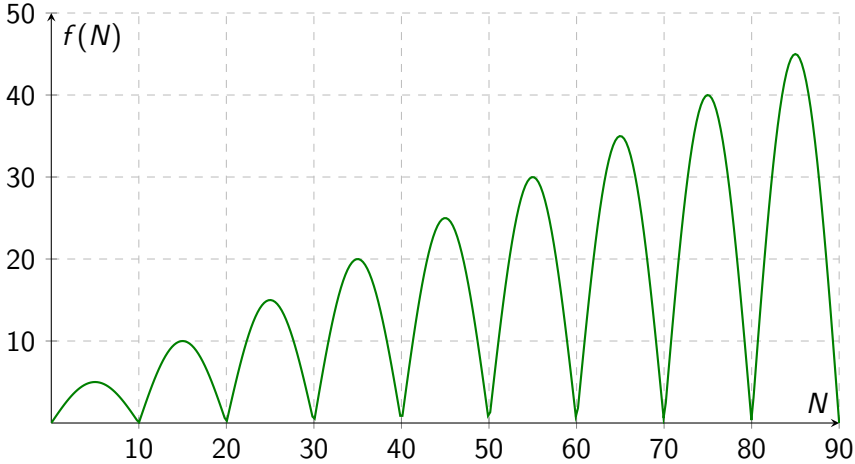
Contents

Asymptotic Analysis

- Big-O Notation (**Upper Bound** $\leq$):
  $f(n) = O(g(n))$ iff $\exists c \quad \exists n_0 \ \forall [n > n_0] \quad f(n) \leq c \cdot g(n)$
- Big-Omega Notation (**Lower Bound** $\geq$):
  $f(n) = \Omega(g(n))$ iff $\exists c \quad \exists n_0 \ \forall [n > n_0] \quad f(n) \geq c \cdot g(n)$
- Big-Theta Notation (**Exact Bound**):
  $f(n) = \Theta(g(n))$ iff $\exists c_1, c_2 \quad \exists n_0 \ \forall [n > n_0] \quad c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$
- Little o Notation (**Strict Upper Bound** $<$):
  $f(n) = o(g(n))$ iff $\forall c \quad \exists n_0 \ \forall [n \geq n_0] \quad f(n) < c \cdot g(n)$
- Little Omega Notation (**Strict Lower Bound** $>$):
  $f(n) = \omega(g(n))$ iff $\forall c \quad \exists n_0 \ \forall [n \geq n_0] \quad f(n) > c \cdot g(n)$

Asymptotic Analysis

Poll (Q1): Let's consider our favourite function once again!



Asymptotic Analysis

Poll (Q1): Select ALL correct statements.
Note: All options use NOT IN ($\notin$).

A. $f(n) \notin \Theta(n)$ ✔
For any c_1, c_2, n_0, find a point $n > n_0$ such that $f(n) = 0$ and clearly $c_1 \cdot n > f(n)$.

B. $f(n) \notin O(n)$
Take $c = 1$ and $n_0 = 10$, we have $f(n) \leq n$.

C. $f(n) \notin \Omega(1)$ ✔
For any c, n_0, find a point $n > n_0$ such that $f(n) = 0$ and clearly $f(n) < c \cdot 1$.

D. $f(n) \notin o(n+1)$ ✔
Note that the $+1$ doesn't change the asymptotic meaning.
Take $c = 0.1$, we can find infinitely many points n such that $f(n) > c \cdot (n+1)$.

Recap: Recurrences

- **Telescoping**: Divide both sides by a term to obtain $\frac{T(n)}{g(n)} = \frac{T(n/b)}{g(n/b)} + h(n)$.
- **Master Theorem**: Given $T(n) = aT(\frac{n}{b}) + f(n)$. There are 3 cases:

	Case	Results
1	$f(n) = O(n^{\log_b a - \varepsilon})$, $\varepsilon > 0$	$T(n) = \Theta(n^{\log_b a})$
2	$f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0$	$T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3	$f(n) = \Omega(n^{\log_b a + \varepsilon})$, $\varepsilon > 0$ **Regularity condition**: $af(\frac{n}{b}) \leq kf(n)$, $k < 1$	$T(n) = \Theta(f(n))$

- **Recursion Tree**: Naively sum up all the work in the tree. Might not work!
- **Substitution Method**: When all the above methods don't work, "guess and check".

Recap: Recurrences

Poll (Q2): Give a tight asymptotic bound for $T(n) = 4T\left(\frac{n}{2}\right) + n \log n$.

A. $T(n) = \Theta(n \log n)$
B. $T(n) = \Theta(n \log^2 n)$
C. $T(n) = \Theta(n^2)$ ✔
D. $T(n) = \Theta(n^2 \log n)$

Solution.
Since $n \log n = O(n^{2-\varepsilon})$ where $\varepsilon = 0.1$, Case #1 of Master Theorem applies.
$\therefore\ T(n) = \Theta(n^2)$.

Given $T(n) = af(\frac{n}{b}) + f(n)$.

1. $f(n) = O(n^{\log_b a - \varepsilon})$, $\epsilon > 0$: $T(n) = \Theta(n^{\log_b a})$
2. $f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0$: $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$, $\epsilon > 0$ **and** $af(\frac{n}{b}) \leq kf(n)$, $k < 1$: $T(n) = \Theta(f(n))$

Recap: Recurrences

Poll (Q3): Give a tight asymptotic bound for $T(n) = 4T\left(\frac{n}{2}\right) + \frac{n^2}{\log n}$.

A. $T(n) = \Theta\left(\frac{n^2}{\log n}\right)$
B. $T(n) = \Theta(n^2)$
C. $T(n) = \Theta(n^2 \log\log n)$ ✔
D. $T(n) = \Theta(n^2 \log n)$

Solution.
Divide both sides by n^2 and apply telescoping, we get $\frac{T(n)}{n^2} = \frac{1}{\log n} + \frac{1}{\log n - 1} + \cdots$. Hence, $T(n) = \Theta(n^2 \log\log n)$ by the harmonic series.

Given $T(n) = af(\frac{n}{b}) + f(n)$.

1. $f(n) = O(n^{\log_b a - \varepsilon})$, $\epsilon > 0$: $T(n) = \Theta(n^{\log_b a})$
2. $f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0$: $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$, $\epsilon > 0$ **and** $af(\frac{n}{b}) \leq kf(n)$, $k < 1$: $T(n) = \Theta(f(n))$

Recap: Recurrences

Poll (Q4): Give a tight asymptotic bound for $T(n) = T\left(\frac{n}{2}\right) + 2T\left(\frac{n}{4}\right) + 2n$.

A. $T(n) = \Theta(n)$
B. $T(n) = \Theta(n \log n)$ ✔
C. $T(n) = \Theta(n^2)$
D. $T(n) = \Theta(n^2 \log n)$

Solution.
Substituting $T(n) = n$, the recursive part sums to n and the merging part is $\Theta(n)$. Therefore, each layer of the recursion tree has balanced work $\Theta(n)$, and there are $\log n$ layers.

Given $T(n) = af(\frac{n}{b}) + f(n)$.

1. $f(n) = O(n^{\log_b a - \varepsilon})$, $\epsilon > 0$: $T(n) = \Theta(n^{\log_b a})$
2. $f(n) = \Theta(n^{\log_b a} \log^k n)$, $k \geq 0$: $T(n) = \Theta(n^{\log_b a} \log^{k+1} n)$
3. $f(n) = \Omega(n^{\log_b a + \varepsilon})$, $\epsilon > 0$ **and** $af(\frac{n}{b}) \leq kf(n)$, $k < 1$: $T(n) = \Theta(f(n))$

Proof of Correctness

- **Iterative Algorithms**: Use a **loop invariant**.
  - Loop invariant = **Expected progress** of the algorithm at a certain snapshot.
  - Descriptive enough to ensure initialization, **maintenance** and termination.
- **Recursive Algorithms**: Proof by **induction** on the problem size.

Lower Bounds (Decision Trees)

Problem: Sorting $n = 3$ distinct numbers by comparing two numbers each time.

Leaves: Output / decision for the input

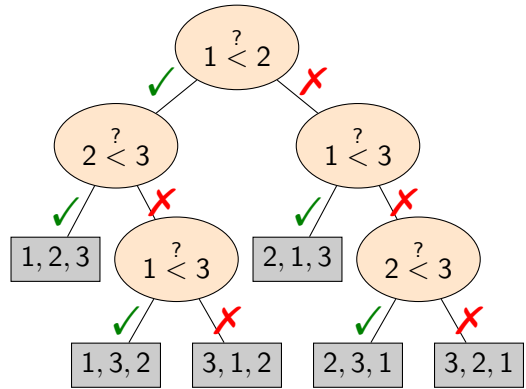
- What are the decisions?
- How many different decisions?

(Internal) Nodes: A comparison

Branches: Outcome of comparison

- What are the outcomes?
- What will be the implication?

Worst Case (number of comparisons) = Height of Tree



Lower Bounds (Decision Trees)

Poll (Q5): There are N students $\{1, 2, \ldots, N\}$ interning at N companies $\{1, 2, \ldots, N\}$. Each student interns at one different company. At least how many Yes/No questions would you have to ask to find out the mapping between the students and the companies?

A. $\Omega(N)$
B. $\Omega(N \log N)$ ✔
C. $\Omega(N^{\log_2 3})$
D. $\Omega(N^2)$

Solution.
There are $N!$ possible mappings.
By decision tree model, we need at least $\log_2 N! = \Omega(N \log N)$ questions.

Average-Case Analysis and Randomized Algorithms

Deterministic Algorithms

- Sometimes, we assume the **input is random**.
- Most of the time, we focus on the **worst case**.

Randomized Algorithms

- Monte-Carlo Algorithm: Probability of success?
- Las-Vegas Algorithm: Expected runtime?
- We focus on the **worst case input**. The randomness comes from the algorithm, not the input.

Randomized Algorithms

Our weapons in analyzing randomized algorithms:

- **Markov's Inequality**: $\mathbb{P}[X \geq k] \leq \frac{\mathbb{E}[X]}{k}$ for any **non-negative** r.v. X.
- **Union bound**: $\mathbb{P}[\bigcup_i X_i] \leq \sum_i \mathbb{P}[X_i]$.
- **Linearity of Expectation**: $\mathbb{E}[X + Y] = \mathbb{E}[X] + \mathbb{E}[Y]$.
- **Probability Amplification**: If each trial succeeds with independent probability p, repeat it k times and it will succeed at least once with probability $1 - (1 - p)^k$.

Randomized Algorithms

Poll (Q6): Consider a random permutation of length n. What is the expected number of indices i such that $P[i] = i$?

A. $1/n$

B. 1 ✔

C. 2

D. None of the other options

Solution.
Let X_i be the indicator random variable that $P[i] = i$.
Since P is a random permutation, $\mathbb{P}[X_i = 1] = 1/n$.
By linearity of expectation, $\mathbb{E}[\sum X_i] = \sum \mathbb{E}[X_i] = n \cdot 1/n = 1$.

Randomized Algorithms

Poll (Q7): Throw N balls into N bins. However, for the i-th ball ($1 \leq i \leq N$), you randomly choose between bins $1 \sim i$ instead of $1 \sim N$. What is the expected number of balls in bin 1?

A. $\Theta(1)$
B. $\Theta(\log N)$ ✔
C. $\Theta(\sqrt{N})$
D. None of the other options

Solution.
Let X_i be the indicator random variable that ball i ends up in bin 1.
$\mathbb{P}[X_i = 1] = 1/i$.
By linearity of expectation, $\mathbb{E}[\sum X_i] = \sum \mathbb{E}[X_i] = 1/1 + 1/2 + \cdots + 1/N = \Theta(\log N)$ (harmonic series).

Randomized Algorithms

Poll (Q8): Let X be a (not necessarily positive) discrete random variable such that $\mathbb{E}[X] = k$. Select ALL correct statements.

A. $\mathbb{P}[X \geq k] > 0$ ✔
There must be a value $\geq$ the expected value (proof by contradiction).

B. $\mathbb{P}[X \leq k] > 0$ ✔
There must be a value $\leq$ the expected value (proof by contradiction).

C. $\mathbb{P}[X = k] > 0$
Not always true. Imagine $X = k - 1$ or $X = k + 1$ each with probability 0.5.

D. $\mathbb{P}[X \geq 2k] \leq 0.5$
Not always true. We need $X \geq 0$ to apply Markov's inequality.

Randomized Algorithms: Freivalds' algorithm

Poll (Q9): You have repeated Freivalds' algorithm 1000 times, and it outputs YES 998 times and NO 2 times. Select the correct statement.

A. It must be the case that $AB = C$.

B. It must be the case that $AB \neq C$. ✔

C. It's possible that $AB = C$ or $AB \neq C$.

D. Contradiction: This outcome is impossible to attain.

Solution.
When $AB = C$, algorithm always output YES (we should not see any NO).
When $AB \neq C$, the algorithm can output anything.

That's it!

Good luck for your midterm!