Midterm cheat sheet

**Module 1 notes:**

**Algorithm analysis**: understand, restate in math, prove correct, analyze speed, code the algorithm.

Order of growth, Lower bounds: remove a positive term from f(n). OR Multiply a negative lower order term in f(n). OR Split a higher order term if needed.

Order of growth, Upper bound: Remove a negative term in f(n). OR Multiply a positive lower order term in f(n).

**Asymptotically efficient**: best for larger inputs.

**L’hospital rule**: What happens to f(n)/g(n) as n🡪infinity,

**Derivatives**:

T(n) = n10 🡪10n10-1

T(n) = log2 n 🡪1/n \* 1/log­­c2 ≈ 1/nfor a some const, c

T(n) = √n 🡪n1/2 = (1/2)n1/2-1=(1/2)n-1/2=1/(2n1/2)=1/(2√n)

**O(f(n))** = if the lim is trends to 0 (Big-O)

Table

Description automatically generated

**Θ(f(n))** = if the lim is const(C) (Theta)

**Ω(f(n))** = if the lim is infinity (Omega)

**Worst and average cases**: both support big-o

**Stirling’s formula**: for n! or n\*(n-1)\*(n-1)…1

n! = (√2πn)(n/e)n for large values of n.

🡪(√n)(nn) simplified

**Loop invariants**: whatever the holder in the loop is holding throughout the entire loop.

**Proving correctness**: base case, inductive case, termination. (start of the loop “holds true”, throughout the loop “true in k iterations”, when the loop is done” loop is finished, same answer”)

**Running time**: time taken for function to complete.

**Order of growth**: what creates the running time.

Log(n3) < log(n!) < 10000(n2) < n2lon(n) < 2n

**Basic operation**: The if in the for loop runs equal amount of time, so pick any of them

**Logarithmic complexity**: the most ideal

**Growth rate**: used to compare two algorithms

**Module 2 notes**:

**Recurrence relations**: recursive part (T(n/2)), nonrecursive(n or c or both). You can drop the lower degree terms (c or n). An equation with the n’th element of a sequence to its predecessors (recursive case)

**Tower of hanoi**: F(n) = 2F(n-1)+1

A picture containing chart

Description automatically generated

Theta(n) = is the constant only if it is linear or a polynomial function.

2log(n): alogcb = blogca  = nlog22 = n1 = n

**Substitution Method**: T(n) = 3T(n/2)+C ; T(1) = c1. Is exponential usually

**Divide and conquer**: quick sort, merge sort, binary search (n log n). Break a problem into not overlapping subproblems. Reduces time complexity usually due to recursion.

Text

Description automatically generated

Text

Description automatically generated

Recurrence tree: for more complex T(n) functions. Is exponential.

Diagram

Description automatically generated

Master’s method: Based on time complexities.

Text

Description automatically generated

**Module 3 notes**:

When using recursion, we solve every subproblem required (not like dynamic programming)

**Time complexity**: is Linear.

**Dynamic programming** is creating an array and using what is already solved (linear F(n)). O(An)

Diagram

Description automatically generated with medium confidence

**Bottom-up**: start with base case and build to solution. Start at bottom right corner of array

**Top-down**: solve bigger problem toward base case. Start at top left corner of array.

**Module 4 notes:**

**Backtracking**: Finds all possible solutions. Identify the parameters that affect the problem, identify subproblems, recursive formula. Find a solution and then go backward to find other options. Generally a slow process (exponential time complexity.

A picture containing text, whiteboard

Description automatically generated

**Module 5 notes / Midterm prep:**

**Back tracking** – gives all possible solutions.

**Dynamic programming** – the most optimal solution always/substructure. Overlapping subproblems.

**Memoization**: store calculated values.

**Greedy**: Hard to design, difficult, efficient compared to dynamic. Best available choice and never looking back

**Knapsack**: cannot use greedy technique due to needing optimal solution. F(x)=max{F[x-wi]+vi}. array domensions = Array[W+1][n+1]

A picture containing application

Description automatically generated

Text, letter

Description automatically generated

Text

Description automatically generated

Text, letter

Description automatically generated

Text

Description automatically generated with medium confidenceTable

Description automatically generated

Table

Description automatically generated

Diagram, schematic

Description automatically generated

Text

Description automatically generated