Mid-term Report - SoS 2020

Topic : Graph Theory Mentor : Tathagat Verma

Mentee : Devansh Jain

Week 1 : 13th April - 19th April
Week 2 : 20th April - 26th April
Week 3 : 27th April - 3nd May

New things learned (These things will be in more detail later)

Proper definitions of Proof, Predicate, Propositions, Axioms

Solved 8-puzzle problem using invariance

It is played on a 3-by-3 grid with 8 square blocks labeled A through H and a blank square. Your goal is to rearrange the blocks so that they are in order. You are permitted to slide blocks horizontally or vertically into the blank square.

Initial	Goal
A B C	A B C
D E F	D E F
H G -	G H -

Find the number of moves required. (Turns out it is impossible)

Solved n-block unstacking game using strong induction

Given a stack composed of n blocks on which we apply the following unstacking recursive algorithm:

i. Split the stack into two smaller stacks of sizes n1 > 0 and n2 > 0 with n1 + n2 = n.

ii. Compute the score of this operation as sum = n1 *n2.

Reiterate (i) and (ii) on the small stacks until you get stuck (cannot split anymore).

Prove that for any strategy the total score would be constant. Find what.

Solved 5 litre and 3 litre bucket puzzle and proved a theorem in general

Suppose you have a ‘a’ litre bottle and a ‘b’ litre bottle (a < b), prove that you can get any volume of water (>=0 and <=b) which is a multiple of gcd(a,b)

Proved Euclid’s Algo for gcd

Modular arithmetic

Euler’s totient function

Fermat’s (little) theorem

Number theory in Encryption:

Turning’s code Version 1 and how to break it

Turning’s code Version 2 and how to break it

RSA

FHE over the integers

Revised following things:

- Algorithmic thinking, peak finding
- Models of computation
- Python cost model
- Sorting techniques: Insertion sort, Merge sort, Heap sort, BST sort, AVL sort
- Counting sort, radix sort, lower bounds for sorting and searching

Graph Algos:

- Breadth First Search
- Depth First Search

Target for 1st three weeks

A. MIT 6.042J Mathematics for Computer Science (Pre-req of MIT 6.006)
   Chapter 1 - 10 (Lectures 1 - 11)

B. MIT 6.006 Introduction to Algorithms
   Units 1 - 6 (Lecture 1 - 18)

C. Introduction to Graph Theory (2nd Edition) - Douglas B. West (For Mathematics of Graph Theory)
   Chapter 1 - 3

What was actually done

A. MIT 6.042J Mathematics for Computer Science (Pre-req of MIT 6.006)
   Chapter 1 - 13 (Lectures 1 - 17)

B. MIT 6.006 Introduction to Algorithms
   Units 1, 2, 5

C. NUS CS3233 - Competitive Programming (For competitive programming)
   Chapter 1, 2

D. Introduction to Graph Theory (2nd Edition) - Douglas B. West (For Mathematics of Graph Theory)
   Chapter 1, 2, (3.i)

Revised PoA

A. http://cp-algorithms.com/
   Problems on graphs algorithms will begin by Week 3 or 4.

B. MIT 6.006 Introduction to Algorithms
   Units 6, 3, 4, 7

C. https://www.geeksforgeeks.org/fundamentals-of-algorithms/ (For Basics of algorithms)

D. https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/ (For concepts of graph algorithms)

E. https://www.youtube.com/watch?v=09_LlHjoEiY&t=7s
   Summary of Graph Algorithms

Original

Following is the original PoA I had.

Short PoA

1. Mathematics revision.
2. Basic algorithms
3. Basic Mathematics of Graph theory
4. Basic graph algorithms like BFS, DFS
5. Shortest path algorithms like Dijkstra, Bellman-Ford
6. A touch of Dynamic Programming
7. Basics of Competitive Programming
8. Coding problems on Graph theory
9. If time permits, then either more rigorous mathematical concepts or more complex algorithms

Programming languages which I may use are C++ and Python.

Detailed PoA

Week 1 : 13th April - 19th April
Week 2 : 20th April - 26th April
Week 3 : 27th April - 3nd May
Week 4 : 4th May - 10th May
Week 5 : 11th May - 17th May

Materials I would be using:

A. MIT 6.042J Mathematics for Computer Science (Pre-req of MIT 6.006)
    a. Part I: Proofs (Week 1)
        i. Chapter 1: Propositions
        ii. Chapter 2: Patterns of proof
        iii. Chapter 3: Induction
        iv. Chapter 4: Number theory
    b. Part II: Structures (Week 1)
        i. Chapter 5: Graph theory
        ii. Chapter 6: Directed graphs
        iii. Chapter 7: Relations and partial orders
        iv. Chapter 8: State machines
    c. Part III: Counting (Week 2)
        i. Chapter 9: Sums and asymptotics
        ii. Chapter 10: Recurrences
        iii. Chapter 11: Cardinality rules (If time permits)
        iv. Chapter 12: Generating functions (If time permits)
        v. Chapter 13: Infinite sets (If time permits)
    d. Part IV: Probability (If time permits)
        i. Chapter 14: Events and probability spaces
        ii. Chapter 15: Conditional probability
        iii. Chapter 16: Independence
        iv. Chapter 17: Random variables and distributions
        v. Chapter 18: Expectation
        vi. Chapter 19: Deviations
        vii. Chapter 20: Random walks

B. MIT 6.006 Introduction to Algorithms
    a. Unit 1: Introduction (Week 1)
        i. Algorithmic thinking, peak finding
        ii. Models of computation, Python cost model, document distance
    b. Unit 2: Sorting and Trees (Week 1)
        i. Insertion sort, merge sort
        ii. Heaps and heap sort
        iii. Binary search trees, BST sort
        iv. AVL trees, AVL sort
        v. Counting sort, radix sort, lower bounds for sorting and searching
    c. Unit 3: Hashing (Week 3)
        i. Hashing with chaining
        ii. Table doubling, Karp-Rabin
        iii. Open addressing, cryptographic hashing
    d. Unit 4: Numerics (Week 3)
        i. Integer arithmetic, Karatsuba multiplication
        ii. Square roots, Newton's method

e. Unit 5: Graphs (Week 2)
    i. Breadth-first search (BFS)
    ii. Depth-first search (DFS), topological sorting
f. Unit 6: Shortest Paths (Week 2)
    i. Single-source shortest paths problem
    ii. Dijkstra
    iii. Bellman-Ford
    iv. Speeding up Dijkstra
g. Unit 7: Dynamic Programming (Week 4)
    i. Memoization, subproblems, guessing, bottom-up; Fibonacci, shortest paths
    ii. Parent pointers; text justification, perfect-information blackjack
    iii. String subproblems, pseudo polynomial time; parenthesization, edit distance, knapsack
    iv. Two kinds of guessing; piano/guitar fingering, Tetris training, Super Mario Bros.
h. Unit 8: Advanced Topics
    i. Computational complexity
    ii. Algorithms research topics

C. https://www.geeksforgeeks.org/fundamentals-of-algorithms/ (For Basics of algorithms)

D. https://www.geeksforgeeks.org/graph-data-structure-and-algorithms/ (For concepts of graph algorithms)

E. Introduction to Graph Theory (2nd Edition) - Douglas B. West (For Mathematics of Graph Theory)
    a. Chapter 1 - Fundamental Concepts (Week 0)
        i. What Is a Graph?
        ii. Paths, Cycles and Trails
        iii. Vertex Degrees and Counting
        iv. Directed Graphs
    b. Chapter 2 - Trees and Distance (Week 1)
        i. Basic Properties
        ii. Spanning Trees and Enumeration
        iii. Optimization and Trees
    c. Chapter 3 - Matching and Factors (Week 2 & 3)
        i. Matching and Covers
        ii. Algorithms and Applications
        iii. Matching in General

    I would be reading other chapters only if time permits.

F. Competitive Programmer's Handbook - Antti Laaksonen (For competitive programming)

G. NUS CS3233 - Competitive Programming (For competitive programming)

H. http://cp-algorithms.com/
   Problems on graphs algorithms will begin by Week 3 or 4.

I. https://www.youtube.com/watch?v=09_LlHjoEiY&t=7s
   Summary of Graph Algorithms