DESIGN AND ANALYSIS OF ALGORITHMS

CS 4120/5120

P, NP, AND FINAL REVIEW

AGENDA

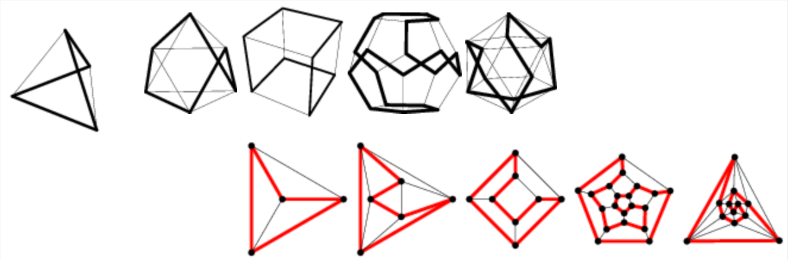
- Polynomial (P), P stands for polynomial
- Non-polynomial (NP)
- Non-polynomial complete (NP-Complete or NPC)
- Final Exam Review

CASE 1 SHORTEST VS LONGEST PATHS

- Given a directed weighted graph $G = (V, E)$ and its weight function $w: E \to \mathbb{R}$.
  - $\mathbb{R}$ is the set of real numbers.
- Problem 1
  - Find the shortest **simple** paths from a source vertex.
  - Can be solved in $O(|V| \cdot |E|)$ time when the graph contains no negative-weighted cycle.
- Problem 2
  - Find the longest **simple** paths from a source vertex.
  - ?

CASE 2
EULER TOUR VS HAMILTONIAN CYCLE

- Given a connected directed graph $G = (V, E)$.
  - Connect means there exist a simple path from any pair of vertices (u, v).
- Problem 1 - ***Euler tour***.
  - Find a cycle that traverses each edge exactly once, although it is allowed to visit each vertex more than once.
  - Solvable in $O(E)$ time
- Problem 2 – Hamiltonian cycle
  - Find a simple cycle that contains each vertex in V.
  - ?



THE CLASSES OF PROBLEMS P, NP, AND NPC

- The class P consists of those problems that are solvable in $O(n^k)$ time for some constant k.
  - All the problems we learned in this semester belong in the class P.
    - Divide and conquer: Maximum-subarray ($O(n \log n)$), Strassen's algorithm ($\Theta(n^{\log 7})$)
    - Prune and search: SELECT ($O(n)$)
    - Sorting algorithms $O(n \log n)$
    - DP: Rod-cutting ($O(n^2)$), matrix-chain multiplication ($O(n^3)$), LCS ($O(n + m)$)
    - Greedy: Activity selection algorithm ($\Theta(n)$), HUFFMAN ($O(n \lg n)$, or depending the implementation),
    - Graph: Graph searching algorithm ($O(|V| + |E|)$), etc…
  - The problem 1 of the previous two cases.

THE CLASSES OF PROBLEMS P, NP, AND NPC

- The class P consists of those problems that are solvable in $O(n^k)$ time for some constant k.
  - Such algorithms are called ***polynomial-time algorithm***.

- Tractable
- Easy

THE CLASSES OF PROBLEMS P, NP, AND NPC

- The **class NP** consists of those problems that are "***verifiable***" in polynomial time.
  - If we were somehow given a "certificate" of a solution, then we could verify that the certificate is correct in time polynomial in the size of the input to the problem.

- Consider the Hamiltonian cycle problem (problem #2 in case 2).
  - Given a directed graph $G = (V, E)$, a certificate would be a sequence $< v_1, v_2, \dots, v_{|v|} >$ of $|V|$ vertices.
  - We can easily check in polynomial time that $(v_i, v_{i+1}) \in E$ for $i = 1, 2, \dots, |V| - 1$ and that $\left(v_{|v|}, v_1\right) \in E$ as well.

THE FAMOUS $P \neq NP$ QUESTION

- Any problem in P is also in NP.
  - If a problem is in P, then we can solve it in polynomial time without the need of a certificate.
  - $P \subseteq NP$

- However, whether a P problem is a proper subset of NP is the open problem.
  - Given a set S, *a proper subset* S' of S means that S' is strictly contained in S and necessarily excludes at least one member of S.

THE CLASSES OF PROBLEMS P, NP, AND NPC

- A problem is in the class NPC if it is in NP and is as "hard" as any problem in NP.
  - The status of NPC class is unknown.

- What does **as "hard" as** mean?

THE CLASSES OF PROBLEMS P, NP, AND NPC

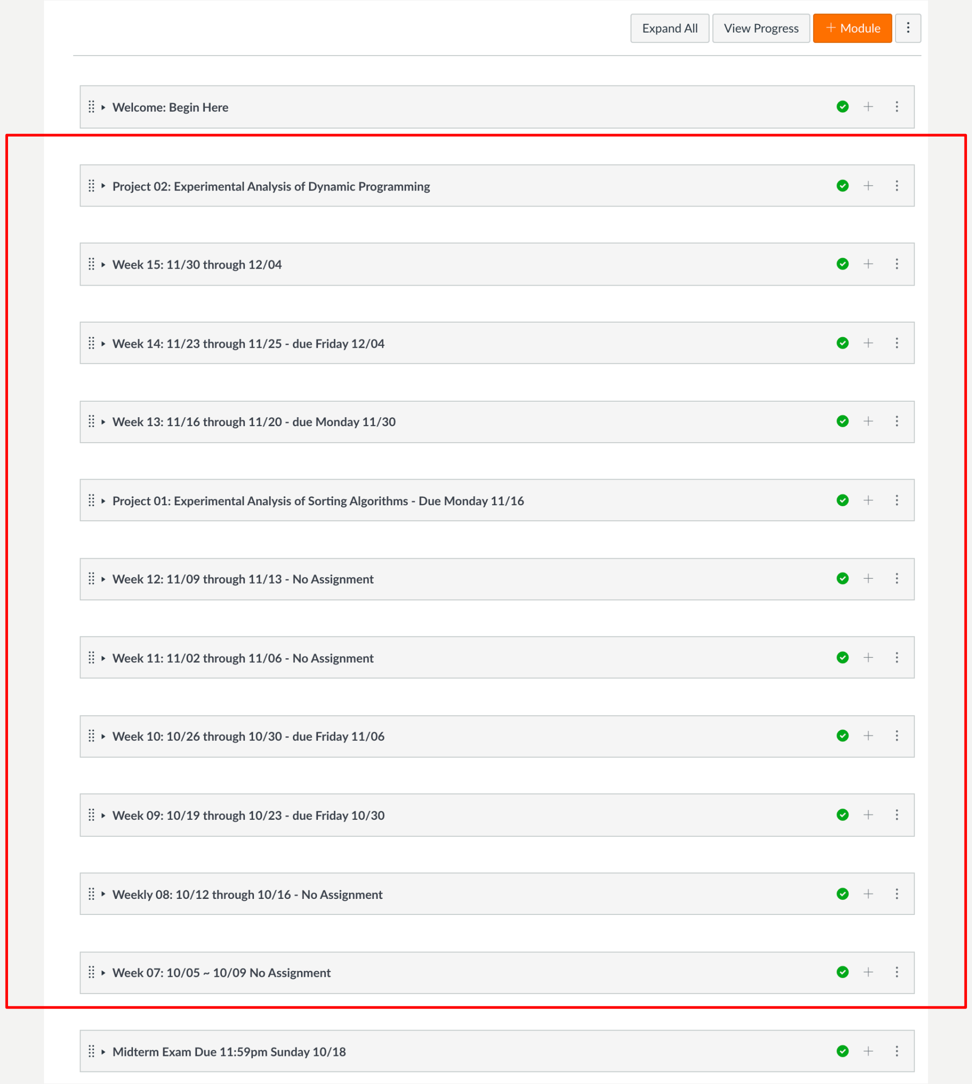
- NPC is the grey area in $P \neq NP$.
  - If we can discover a polynomial-time algorithm for an NP-complete problem, then the problem becomes P.
  - If we can prove that there exist no polynomial-time algorithm for an NP-complete problem, then we might be able to find a certificate to verify that the problem is NP.
  - No polynomial-time algorithm has yet been discovered for an NP-complete problem, nor has anyone yet been able to prove that no polynomial-time algorithm can exist for any one of them.
    - This is why we have a class NPC that describe such problems.

NEXT UP
FINAL REVIEW

- Available at 10 am on Monday, December 7
- Close at 11:59 pm on Friday, December 11
- Two attempts with each being limited by 2.5 hours.
- Scores of the two attempts will be averaged out.
- **INDIVIDUAL WORK**
- You may use notes. Manage your time wisely.

TOPICS

- Week 07 till TODAY.
- Review tips
  - Assignments in the weekly module.



SELECTION PROBLEM

- RANDOMIZED-SELECT
  - PARTITION, RANDOMIZED-PARTITION
  - Best, worst-case scenario running time
- SELECT in linear time in the worst-case scenario
  - Five steps
  - Divide into groups of five elements, …
  - Think about dividing into groups of 7 elements

SORTING

- Quicksort
  - PATITION
  - RANDOMIZED-PARTITION
  - Best-case, worst-case scenario running time
- Heapsort
  - Data structure: heap, max-heap properties, min-heap properties
  - BUILD-MAX-HEAP
  - MAX-HEAPIFY
  - HEAPSORT

DYNAMIC PROGRAMMING

- Key ingredients
- Elements
- The optimal substructure (notation, proof), the recursive formula of computing the optimal value, and the running time of the following
  - Rod-cutting
  - Matrix-chain
  - Longest-common subsequence

GREEDY STRATEGY

- Key ingredients
- Activity selection problem
  - Optimal substructure (notation, proof)
  - The greedy property (notation, proof)
  - Running time
- HUFFMAN algorithm
  - Variable-length codeword
  - Prefix codes
  - Data structure
  - Running time

GRAPH ALGORITHMS

- Graph representation
- Graph searching algorithms
  - BFS
    - Strategy, data structure, object, result, running time
  - DFS
    - Strategy, object, result, running time

GRAPH ALGORITHMS

- Shortest-paths problem
  - ***Single-source*** shortest-paths problem
    - Can be used to solve many variants of shortest-paths problem
  - Dijkstra’s algorithm
    - Directed, weighted graph, with no negative edge
    - Strategy, data structure, object, running time
  - Bellman-ford
    - Directed, weighted graph with negative edge.
    - Strategy, result, running time

REFERENCE