CSC 349-03: Design and Analysis of Algorithms **Spring 2022**

Lecture 1: May 13th, 2022

Instructor: Theresa Migler *Scribe: (Hank Tsai)*

1 Announcements

Announcements as of May 13th

Dr. Migler wasn't feeling well, so class is canceled and lecture recording was given, she also reminded us to stay healthy and kind.

2 Complexity

2.1 Decision Problem

Definition 1: A Decision Problem is a problem for which any proposed solution can be quickly checked for correctness.
Key Idea: Problem, Quickly
Explanation: Checking(I, S), in polynomial time return yes/no
There exists a "checking algorithm" C, that takes as an input
*The given instance of the problem, I
*The proposed solution, S
C outputs true if and only if S is a solution for Instance I. Further, the running time of C on instance (I, S), is polynomial in I
I think of C as a grading algorithm

2.2 NP

Definition 2: The class of all decision problems is denoted by NP
Examples: Knapsack
It's an optimization problem, and we convert it into a decision problem by introducing a threshold and check that your solution meets that threshold.
Knapsack optimizations: return the set of items with max value subject to the weight constraint.
Summation of wi less than or equal to W
Summation of vi greater than or equal to k
Other Examples: Independent Set, Travelling Saleperson Problem, Minimum Spanning Tree, Matching

2.3 P

Definition 3: P is the class of all decision problem that can be solved in polynomial time.
Keyword: can be solved
Example of decision problems in P:
Minimum Spanning Tree
Longest Increasing Subsequence
Independent Set On Trees
Bipartite Matching
Many, Many others

2.4 Relation between P and NP

P is either a strict subset of NP: meaning that each decision problem that can be checked in polynomial time cannot necessarily be solved in polynomial time

P is equal to NP: meaning that decision problems with polynomial time checking algorithm will have polynomial time solving algorithm

Note: We don't know which set we are in

Are there any decision problems that couldn't be solved in polynomial time?
If so: P Not Equal to NP
If not: P Equal to NP
The answer is unknown. Most mathematicians believe that P is Not equal to NP

2.5 NP-Complete

Definition: A problem X is NP complete if
1.X is in NP and
2. every problem in NP is reducible to X in polynomial time
Note: A problem satisfying condition 2 is NP-Hard if it doesn't satisfy condition 1
If any problem in NP-Complete can be solved in Polynomial time, all other NP problems can also be solved in polynomial time, which means P = NP

Historical Context: Levint Cook came up with SAT hard problem in 1971
Later, Richard Karp from UC Berkeley came up with 21 other problems that are at least as hard as SAT

2.6 Reduction

Definition: A reduction from a decision problem A to a decision problem B (A transforms to B) is a polynomial time algorithm, f:
*That transforms an instance I, of A into an instance f(I), of B.
Together with another polynomial time algorithm, h:
*That maps any solution S of f(I) back into a solution h(s) of I
If f(I) has no solution, then neither does I

Instance I - f - f(I) - Algorithm B - Solution S of f(I) - Solution h(s) of I
Instance I - f - f(I) - Algorithm B - No Solution of f(I) - No Solution to I

Conclusion: If we can come up with f and h algorithm, Problem B is at least as hard as Problem A