# HKN CS 374 Final Review

Timothy Klem
Mahir Morshed
6 May 2018

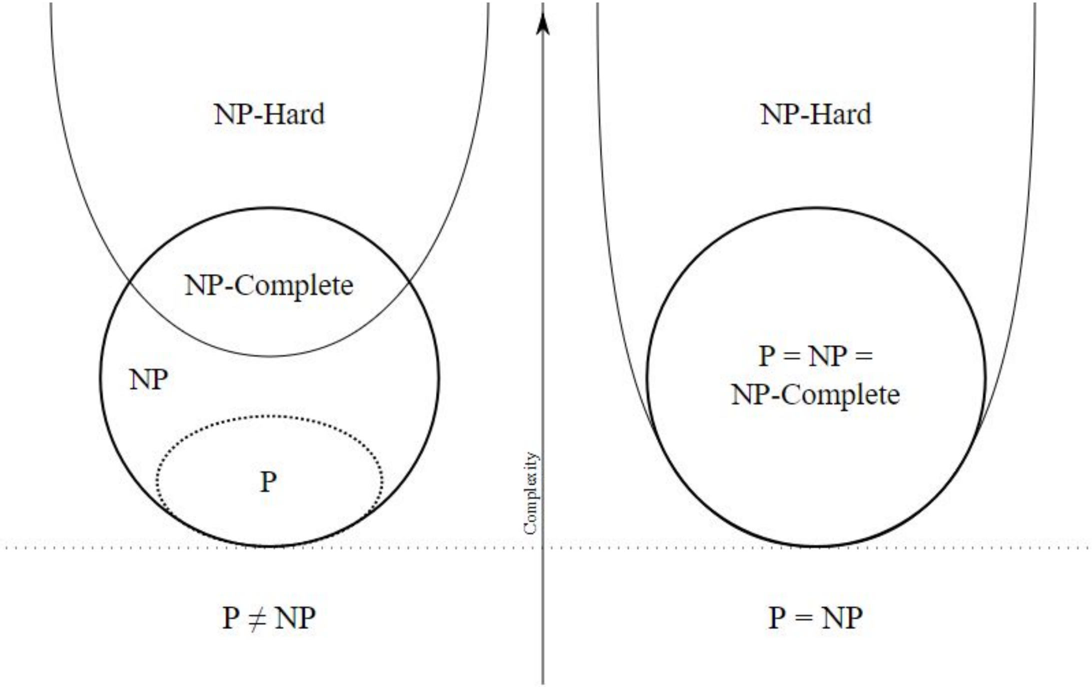
# Topic Outline

- P vs. NP
- Undecidability
- NP-complete reductions
    - Graph gadgets
    - Input transformations

# P vs. NP

- P - problems that can be *solved* in polynomial time
- NP - problems that can be *verified* in polynomial time
- NP-hard - problems that are at least as hard as NP problems
  - i.e. all NP problems can be reduced in polynomial time to NP-hard ones
- NP-complete - problems that are NP *and* NP-hard

# P vs. NP


NP-Hard
NP-Complete
NP
P
P ≠ NP
Complexity
NP-Hard
P = NP =
NP-Complete
P = NP

# Undecidability

- Recognizable (recursively enumerable) - there exists a TM that can accept and halt all of the strings in the language, but we're not sure what happens when the TM runs on other inputs
- Decidable - there exists a TM that can accept and halt on strings within the language, and reject and halt on strings not in the language
- Languages constructed from a TM ( L(M) ) are defined as a set of all strings accepted by a TM

# True/False - Decidability and Recognizability

(A) The language $A_{TM}$ is recognizable.
(B) The complement of $A_{TM}$ is recognizable.
(C) If $A$ reduces to $B$ and $A$ is decidable, then $B$ is decidable.
(D) If $L \subset \{0\}^*$ then $L$ is decidable.
(E) If $L$ reduces to $\{0^n 1^n | n \geq 0\}$ then $L$ is decidable.
(F) If a problem is undecidable, then it can also be NP-HARD.
(G) If $L_1$ is decidable and $L_2$ is recognizable, then $L_1 - L_2$ is recognizable.
(H) If $L_1$ is recognizable and $L_2$ is decidable, then $L_1 L_2$ is recognizable.
(I) If $L_1$ is recognizable and $L_2$ is unrecognizable, $L_1 \cap L_2$ is unrecognizable.
(J) The language $\{\langle M \rangle | M$ is a TM and $L(M)$ is countable$\}$ is decidable.

Consider a conjunctive normal form (CNF) formula $F$ with all clauses being of size 2, except for 10 clauses that are of size at most 7 (i.e., these clauses are made out of up to seven literals). Consider the problem of deciding if such a formula is satisfiable. Assuming $P \neq NP$, this problem is in $NP$ but not in $P$.

Say someone discovers two algorithms $A_Y$ and $A_N$. Both algorithms read an undirected graph $G$ and a number $k$. If $G$ has an independent set of size $k$, then $A_Y$ would stop in polynomial time and output YES, but $A_N$ might run forever. Similarly, if $G$ does not have an independent set of size $k$, then the algorithm $A_N$ would stop in polynomial time and output NO, but $A_Y$ might run forever. If this scenario were to occur, then we can conclude that $P = NP$.

# NP-hard reductions

- Treat your algorithm as a black box
- Constrain the input into the black box so a Yes/No from the black box leads to a Yes/No answer to a known NP-hard problem
  - Constraining input usually involves encoding (booleans to real scenarios) or graph gadgets (cliques, stars, lines, etc.)
- You might need to use the black box more than once
- Any reduction must run in polynomial time
- To prove NP-completeness, must show how to verify a solution in polynomial time
- Any input transformation must be justified with a two-way proof

12. Prove that the following problem (which we call MATCH) is NP-hard. The input is a finite set $S$ of strings, all of the same length $n$, over the alphabet $\{0, 1, 2\}$. The problem is to determine whether there is a string $w \in \{0, 1\}^n$ such that for every string $s \in S$, the strings $s$ and $w$ have the same symbol in at least one position.

For example, given the set $S = \{01220, 21110, 21120, 00211, 11101\}$, the correct output is TRUE, because the string $w = 01001$ matches the first three strings of $S$ in the second position, and matches the last two strings of $S$ in the last position. On the other hand, given the set $S = \{00, 11, 01, 10\}$, the correct output is FALSE.

*[Hint: Describe a reduction from SAT (or 3SAT)]*

Recall that a spanning tree $T$ of a graph $G$ with $n$ vertices is a subgraph of $G$ with $n-1$ edges that contains all $n$ vertices of $G$. Given that SPAN2 is NP-hard, prove that SPAN374 is also NP-hard.

*SPAN2: Given an undirected graph G, does G contain a spanning tree in which every node has degree at most 2?*

*SPAN374: Given an undirected graph G, does G contain a spanning tree in which every node has degree at most* ***374****?*