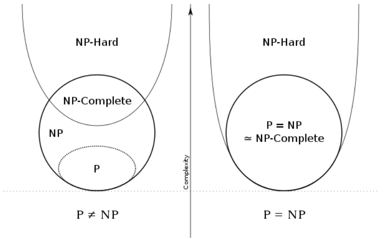
Note: Your TA probably will not cover all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. They are deliberately made long so they can serve as a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

Last week we learned that if there exists a polynomial reduction from problem A to problem B, problem B is at least as hard as problem A. From this, we can define **complexity classes** which essentially gauge the "hardness" of problems:

- **P**: a decision problem that can be solved in polynomial time.
- **NP**: a decision problem for which a potential solution can be verified in polynomial time.
- **NP**-Hard: a decision problem that all problems in **NP** can reduce to.
- **NP**-Complete: a decision problem that is both in **NP** and in **NP**-Hard.

We can organize these complexity classes graphically as follows:



Proving NP-Completeness

To prove a problem is **NP**-Complete, you must prove the problem is in **NP** and it is in **NP**-Hard. In other words, you must show that (1) there exists a polynomial verifier, and (2) there exists a reduction from any **NP**-Complete problem to the problem.

1 P or NP, that is the question

For the following questions, circle the (unique) condition that would make the statement true.

(a) If B is **NP**-complete, then for any problem $A \in$ **NP**, there exists a polynomial-time reduction from A to B.

Always True True iff **P** = **NP** True iff **P** ≠ **NP** Always False

(b) If B is in **NP**, then for any problem $A \in$ **P**, there exists a polynomial-time reduction from A to B.

Always True True iff **P** = **NP** True iff **P** ≠ **NP** Always False

(c) 2 SAT is **NP**-complete.

Always True True iff **P** = **NP** True iff **P** ≠ **NP** Always False

(d) Minimum Spanning Tree is in **NP**.

Always True True iff **P** = **NP** True iff **P** ≠ **NP** Always False

2 NP Basics

Assume A reduces to B in polynomial time. In each part you will be given a fact about one of the problems. What information can you derive of the other problem given each fact? Each part should be considered independent; i.e., you should not use the fact given in part (a) as part of your analysis of part (b).

1. A is in **P**.
2. B is in **P**.
3. A is **NP**-hard.
4. B is **NP**-hard.

3 Breaking Encryption

After years of research, Horizon Wireless released an encryption algorithm E that encrypts an n-bit message in time $O(n^2)$. Show that if $\mathbf{P} = \mathbf{NP}$ then this encryption algorithm can be broken in polynomial time. More precisely, argue that if $\mathbf{P} = \mathbf{NP}$, then the following decryption problem can be solved in polynomial time.

Decrypt :
Input: An encrypted message M (encrypted using the algorithm E)
Goal: Decryption of M.

Hint: suppose you can solve Decrypt *using the decryption algorithm D. What complexity class is D in?*

4 Hitting Set

In the Hitting Set Problem, we are given a family of finite integer sets $\{S_1, S_2, \ldots, S_n\}$ and a budget b, and we wish to find an integer set H of size $\leq b$ which intersects every S_i, if such an H exists. In other words, we want $H \cap S_i \neq \emptyset$ for all $i \in \{1, \ldots, n\}$.

Show that the Hitting Set Problem is **NP**-complete. *(Hint: Hitting Set generalizes one of the problems covered in Chapter 8 of the textbook.)*

5 Reliable Network

We define the Reliable Network problem as follows. We are given a cost matrix $c_{ij} \in \mathbb{R}^{n \times n}$, a connectivity requirement matrix $r_{ij} \in \mathbb{R}^{n \times n}$, and a budget $b \in \mathbb{R}$. Our goal is to find a graph $G = (\{1, ..., n\}, E)$ such that the total cost of all edges is b or less and between any two distinct vertices i and j there are r_{ij} vertex-disjoint paths. Show that Reliable Network is **NP**-Complete.