Ch 8.3 Efficient Certification and Def. of NP

- Finding a solution vs. Checking a (proposed) solution
- Decision Problems & P vs. NP
  - P: Class of decision problems that can be solved in poly-time
  - NP: Class of decision problems that can be checked in poly-time (for all yes-instances(*))

Ch 8.3 Efficient Certification and Def. of NP

- Problems & Algorithms
  - Input: a finite (binary) string ***s*** with length |***s***|
  - A decision problem ***X***: The *set of strings* on which the answer is “yes”
  - An algorithm ***A***: ***A***(***s***) is the returned value by ***A*** on input ***s*** (which is either “yes” or “no”)
  - A *solves* ***X*** if for all strings ***s***, ***A***(***s***) = “yes” if and only if ***s*** is in ***X***.
  - ***A*** has a polynomial running time if ***A*** terminates on ***s*** in at most O(***f***(|***s***|)) steps for all ***s*** where ***f*** is a poly-function.
- Class ***P***: The set of decision problems X for which there exists a poly-time algorithm that solves X

Ch 8.3 Efficient Certification and Def. of NP

Eg. NP: Problem Composite: {4, 6, 8, 9, 10, ...}
s: 437669
t: 541

i.e
```
Algo -
{if (437669 % 541 == 0)
    return yes;
else:
    false; }
```

- *Efficient Certifier* for Problem X
  - Input: a finite (binary) string ***s*** with length |***s***|
  - A decision problem ***X***: The set of strings on which the answer is “yes”
  - An efficient certifier ***B*** is a poly-time algorithm that takes ***s*** (input) and ***t*** (proof) AND there exists a polynomial function ***p*** so that for every string ***s***, we have ***s*** in ***X*** if and only if there exists a string ***t*** such that |***t***| <= ***p***(|***s***|) and ***B***(***s***,***t***) = “yes”.

    't' is a proper ans to 's'
  - B does NOT solve X but it can ***check*** if ***s*** is really a yes-instance (with the help of ***t***).

Ch 8.3 Efficient Certification and Def. of NP

- Class *P*: Set of decision problems X for which there exists a poly-time algorithm that solves X.
- Class *NP*: Set of decision problems X for which there exists a poly-time certifier for X.
  - Max-flow, MST, Min-cut, and WIS
  - Independent Set, Vertex Cover, and 3-SAT

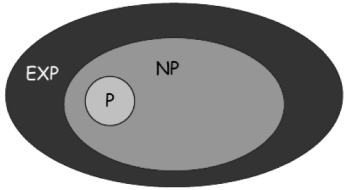
- Lemma (8.10): P is a subset of NP.
  - Given any problem X in P, we can design B by just solving X directly (and thus “an empty string” would be a sufficient proof).

- To prove that problem X is in NP:
  - Show that X is in P (by designing a poly-time algorithm).
  - Or, show that there exists a short proof for yes-instances & there exists a verifier that can check correctness of such proofs in poly-time.

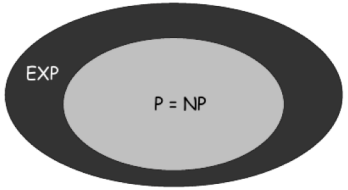
The Main Question: P Versus NP

Does P = NP? [Cook 1971, Edmonds, Levin, Yablonski, Gödel]

- Is the decision problem as easy as the certification problem?
- Clay $1 million prize.



If P ≠ NP



If P = NP

would break RSA cryptography (and potentially collapse economy)

If yes: Efficient algorithms for 3-COLOR, TSP, FACTOR, SAT, ...

If no: No efficient algorithms possible for 3-COLOR, TSP, SAT, ...

Consensus opinion on P = NP? Probably no.

Required Readings & Exercises

- KT Chapters 8.1, 8.2, and 8.3.
- KT Chapter 8.10
  - Try to prove that each problem listed in KT Chapter 8.10 is in NP.
  - NP-completeness & NP-hardness will be discussed in L32.

- KT Exercise 8.1