CS 173 Lecture 18b: NP

P : (yes/no) problems solvable in $O(n^k)$ time for some $k \in \mathbb{R}$

EXP: solvable in $2^{O(n^k)}$ time

NP : (yes/no) problems whose <u>yes</u> solutions are <u>verifiable</u> in $O(n^k)$ time for some $k \in \mathbb{R}$.

Verifiable?

Given a graph G, & # k, is $\chi(G) \leq k$?

(yes)
→ Solution: a k-coloring of G.

often called proof, witness, certificate

Alg for verification:

```
for each edge uv,
    if color(u) = color(v)
        return false
return true
```

$O(n^2)$ time.

What about computing a (yes) solution?

To the best of our knowledge, is <u>no</u> polynomial time alg.

$P \subseteq NP$:

for problems in P, <u>compute</u> a proof in polynomial time

$NP \subseteq EXP$:

if "solution" can be verified in $O(n^k)$ time,

there are at most $2^{O(n^k)}$ possible "solutions"

Why is NP important?

Many examples of problems in NP − P

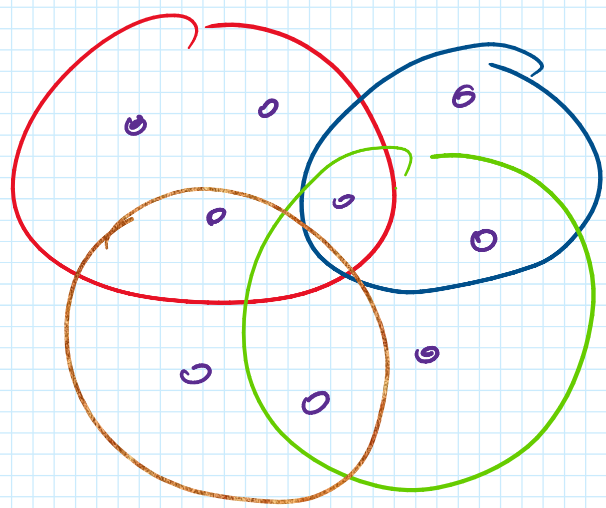
job scheduling: given jobs that take time $t_1 \ldots t_n$, resp. can we schedule jobs over m workers so that the last job finishes before time T?

proof: schedule

graph coloring

resource allocation (many variants)

e.g. set cover / hitting set



Can we pick $\le k$ things so that everybody is satisfied

circuit satisfiability:

given a circuit of {boolean}

$$(p_1 \wedge p_2) \vee (p_3 \wedge \neg p_4 \wedge p_5)$$

Q: Is there a setting to p_i's so that the circuit evaluates to TRUE?

proof: setting to the p_i's

the circuit evaluates to TRUE?

proof: setting to the p_i's

NP-completeness

informally: $Q \in$ NP-Complete iff

$\exists$ poly-time alg for Q

$\longrightarrow \exists$ poly-time alg for all $Q' \in NP$.

"the hardest problems in NP"

P vs NP: one of most important questions in Math / CS.

NP-complete: scheduling, graph coloring (for $k \geq 3$) circuit satisfiability.

common assumption: $P \neq NP$

believed to be in NP − NP-complete

graph isomorphism.

as of 2017 $\exists$ "quasi polynomial" time alg.

primality testing

easy alg: $O(\sqrt{n})$ but n can be written in $O(\log n)$ bits.

input size.

"randomized": $O(1)$, correct "w/ high probability"

$\rightarrow$ many believe $\exists$ alg for testing primality of n in $O((\log n)^k)$ time.

beyond NP-completeness:

results of the form can't solve in $O(n^l)$ time for $l < k$

but can verify in $O(n^l)$ time.