CS124 Lecture 19

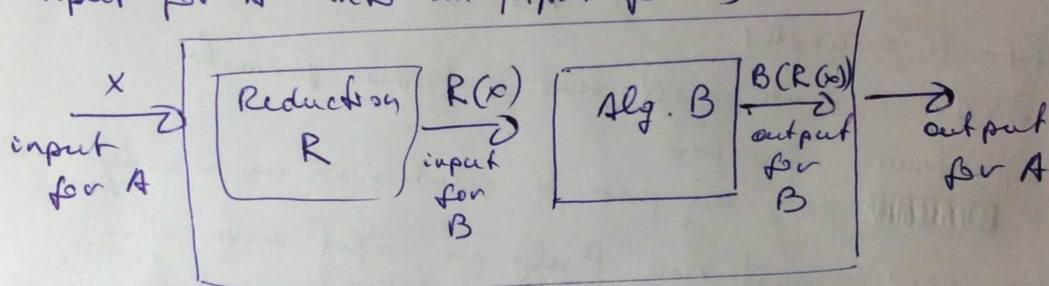
Polynomial time problems P

P = set of problems with a yes/no answer s.t.
∃ algorithm A, integer k s.t.
A runs in $O(n^k)$ time on inputs of size n

Luckily, most algs have small polynomials. [redacted]

Reduction

Poly-time Reduction from A to B
is an Alg. R which turns
input for A into an input for B



Polytime Alg for B
Polytime reduction
=> Polytime Alg for A

$A \leq_R B$

$T(x)$ = time $x \to R(x)$
$S(y)$ = time for Alg. B on y
$T(x) + S(|R(x)|) = \text{poly}_n(|x|)$

$A \leq_R B,\ B \leq_R C \Rightarrow A \leq_R C$
composition of reductions, transitivity

$$x \to R_1(x) \to R_2(R_1(x))$$

⟶ direction of reduction
$A \leq_R B$

① B easy => A easy, use reduction to solve probs.

② A hard => B hard, use reduction identify hard problems

NP (nondeterministic polynomial time)

"checked" in polynomial time

Problems where – if the answer is yes
there is a "short" certificate, s.t.
poly-size

given the soln. / short certificate
the soln. can be verified in poly time

$\exists$ a checking algorithm that
takes as input : problem input, short certificate
and returns : valid, if answer is yes to problem
and certificate is valid

no otherwise

certificate examples

- 3SAT : truth asst that satisfies formula
- Compositeness : factor
- [redacted] poly-size path to solution in an non-deterministic tree

$P \subseteq NP$ just run P alg. on input with empty string certificate

$P \stackrel{?}{=} NP$

Hardest Problems in NP

NP – Complete :

① in NP

② (NP – Hardness) all other problems in NP reduce to it

defn: $\underset{NP}{A} \le_R \underset{\text{NP-complete}}{B}$

may not be in NP

Once a problem is NP – complete

X, new NP – complete problem

show $X \le_R Y$

$\in NP$

Cook-Levin theorem (NP-Complete)

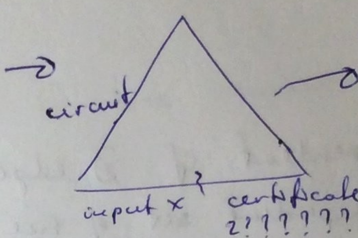
Circuit SAT:
given a Boolean circuit, values of some inputs,
can the rest of the inputs be set s.t.
it evaluates to true

Iff a problem is in NP, it can be reduced to circuit SAT.

[input, certificate] Alg. A

Alg A = Poly sized boolean circuit

Alg A, input →



→ solution to circuit SAT
solves for certificate that makes x evaluate to true
↓
if certificate found → true
if no certificate found → false

⇒ any NP problem can be reduced to circuit SAT

3SAT is NP-Complete (Circuit SAT $\le_R$ 3SAT)

Reduction circuit SAT to 3SAT formula, IFF ✓
Poly-time transformation ✓

① input gates: if T,	include (x)
if F,	include $(\bar{x})$
unspec. input,	nothing
② $x = y \vee z$	include $(\bar{y} \vee x) \wedge (\bar{z} \vee x) \wedge (\bar{x} \vee y \vee z)$
$x = y \wedge z$	include $(\bar{x} \vee y) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee \bar{z} \vee x)$
$x =$ NOT y	incl. $(\bar{x} \vee \bar{y}) \wedge (x \vee y)$
③ output gate x	(x)

$3SAT \leq_P ILP$

T ⟷ [redacted] 1 $\qquad 0 \leq x \leq 1$

F ⟷ [redacted] 0 $\qquad$ for all variables in 3SAT clauses

$(x \vee \bar{y} \vee z) \wedge (a \vee b \vee c) \ldots$

⇓ IFF $\qquad\qquad$ ⇓ IFF

$x + (1-y) + z \geq 1 \qquad a + b + c \geq 1$

$3SAT \leq_P$ Independent Set

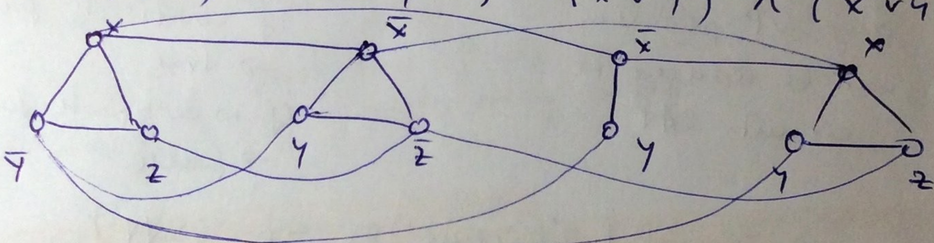
Independent Set

$G = (V, E)$

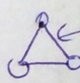
$I \subseteq V$ is independent if $\nexists$ edge (u, v), s.t. $u \in I, v \in I$

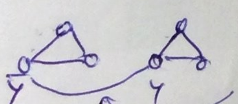
Is there an independent set of size $\geq k$

$(x \vee \bar{y} \vee z) \wedge (\bar{x} \vee y \vee \bar{z}) \wedge (\bar{x} \vee y) \wedge (x \vee y \vee z)$



formula satisfiable ⟺ ∃ indep. set of size
$\geq$ # of clauses


← force to pick <u>one</u> vertex from each clause
↳ true vertex



cannot pick $\bar{y}$ and y, both cannot be true

poly-time construction ✓

Independent set $\leq_R$ Vertex Cover

$S \subseteq V$ s.t. all edges are incident to at least one vertex in S

"vertices covering all edges"

Is there a VC $\leq$ [redacted] size k

S is VC $\Leftrightarrow$ $V - S$ is indep. set

Independent set $\leq_R$ Clique

take G^c

independent sets in G are cliques in G^c