12: NP Complete Problems

Friday, April 30, 2021 11:20 AM

NP: Big Idea → $\overset{x}{\text{problems}}$ where you can $\overset{\text{in polynomial time}}{\boxed{\text{easily}}}$ $\overset{M}{\text{check}}$ a $\overset{\text{(solution)}\ y}{\text{strategy}}$

(defn doesn't specify whether finding a good strategy is hard. Could be easy or hard)

ex: Sudoku, 3-SAT, Hamiltonian Path,

Proof:

- Description of how you check strategy (make sure you'll catch bad strategies)!
- Analyze runtime of checks.
- Analyze length of strategy string

} Polynomial time strategy

] Proof convinces you can easily check (Some problems, like Halting Problem, can't be easily checked!)

Connection to Definition:

NP = Non-deterministic Polynomial Time

A problem is in NP if

- Yes-No Problem
- ∃ M, a $\boxed{\text{polynomial time}}$ algorithm s.t. (verifier)
  - If instance x has output "Yes" → ∃ y (witness) s.t. $\boxed{|y| = O(\text{poly}(|x|))}$ $\boxed{M(x,y) \text{ outputs } 1}$
  - If instance x has output "No" → ∀ y $\boxed{M(x,y) \text{ outputs } 0}$

Hamiltonian Path ∈ NP

Not necessary, just helpful

Pf: [Interpret y as sequence of edges. x is description of $G=(V,E)$, $|V|=n$

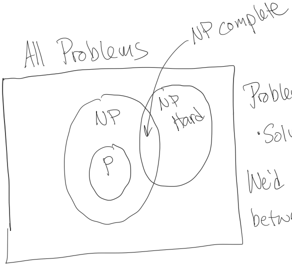
Let $M(x,y)=1$ iff:

$\boxed{\text{Runtime:}}$

Checks

- y is path of length n-1 — $O(n)$
- y only contains edges in E — $O(n^3)$
- y starts at s, ends at t. — $O(1)$
- y encounters each vertex 1 time — $O(n^2)$

$\boxed{|y| = O(n)}$


All Problems
NP complete
NP
P
NP Hard

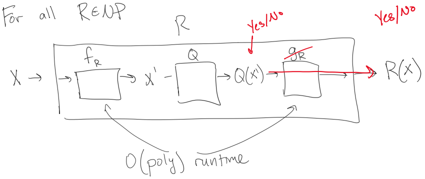
Problems in P are also in NP.

- Solution can be easily checked (and found!)

We'd like a way to distinguish between easy and hard problems in NP
↓ P   ↓ likely not in P

NP-Hard: Q ∈ NP-Hard if ∀ R ∈ NP, $R \le_{poly} Q$.

For all R ∈ NP


R
Yes/No
Yes/No
X
f_R
x'
Q
Q(x)
~~g_R~~
R(x)
O(poly) runtime

Q is so hard/powerful it lets you solve anything in NP

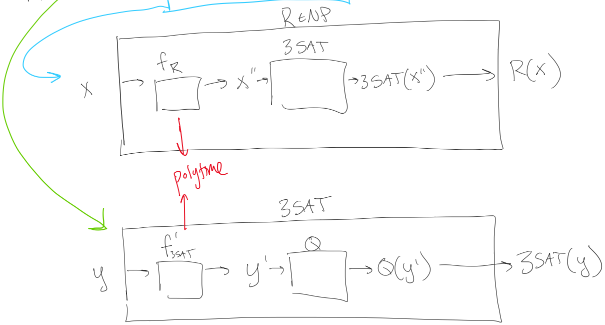
NP-Complete: Q ∈ NP-Complete iff

- Q ∈ NP [Know how to prove]
- Q ∈ NP-Hard [? Proof ?]

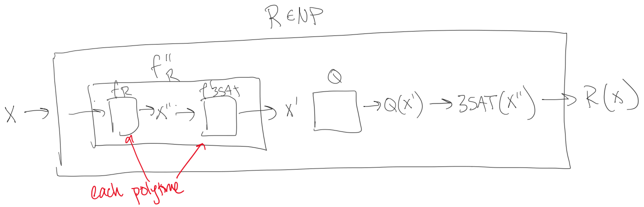
How → Show $3\text{-SAT} \le_p Q$ Why?

Theorem: · If $3SAT \le_p Q$, then Q ∈ NP-Hard:

Pf: · Fact: 3-SAT is NP-Hard (See 301)


R ∈ NP
3SAT
x
f_R
x''
3SAT(x'')
R(x)
polytime
3SAT
y
f'_{3SAT}
y'
Q
Q(y')
3SAT(y)

Then


R ∈ NP
x →
f''_R
f_R
x''
f_3SAT
x'
Q
Q(x')
3SAT(x'')
R(x)
each polytime

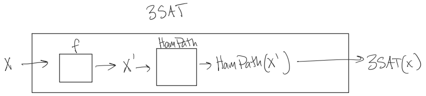
So Q is NP-Hard

Theorem: Hamiltonian Path is NP-Complete

Pf • Hamiltonian Path ∈ NP (see prev. proof)

- Hamiltonian Path ∈ NP-Hard

↳ Show 3-SAT $\leq_p$ HamPath ↓

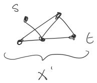

3SAT
x
f
x'
HamPath
HamPath(x')
3SAT(x)

↓ Need to prove this reduction exists

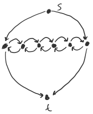
1. Describe f
2. Analyze runtime of f (polynomial)
3. Prove HamPath(x') = Yes iff 3SAT(x) = Yes (correct output!)
    a) If HamPath(x') = Yes, 3SAT(x) = Yes
    b) If 3SAT(x) = Yes, HamPath(x') = Yes

1. Describe reduction from 3SAT to HamPath

$\underbrace{(z_1 \vee \bar{z}_3 \vee z_6) \wedge \ldots}_{x} \Rightarrow$



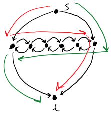
Idea: Use
Graph Gadget (a little graph that implements some functionality)

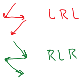


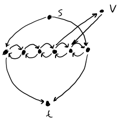
How many Hamiltonian Paths are there from s to t?

A) 0
B) 1
C) 2
D) >2

2 paths



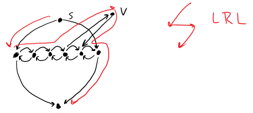




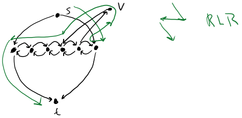
How many Hamiltonian Paths are there from s to t?

A) 0   C) 2
B) 1   D) >2

1 path

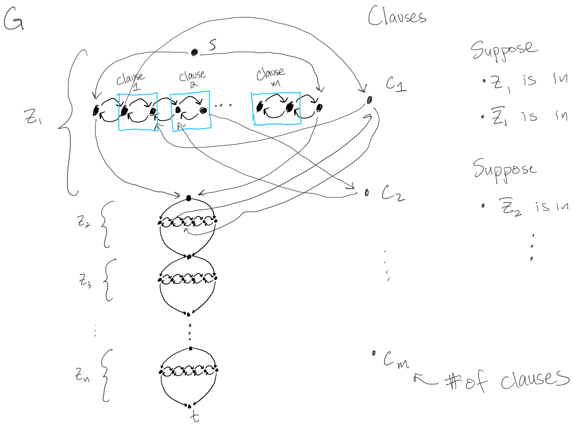


Reverse direction of edges to v:



One gadget for each variable z_i:

- LRL path ⟷ z_i = True
- RLR path ⟷ z_i = False


G
Clauses
s
clause 1
clause 2
clause m
z_1
c_1
c_2
z_2
z_3
z_n
c_m
# of clauses
t

Suppose

- z_1 is in clause 1
- $\bar{z}_1$ is in clause 2

Suppose

- $\bar{z}_2$ is in clause 1

⋮

(Visual description of reduction)

ex: $X = (z_1 \vee z_2) \wedge (\neg z_1 \vee \neg z_2)$

3-SAT(x) = Yes if ∃ a satisfying assignment. No otherwise

3SAT(x) = Yes b/c

satisfies

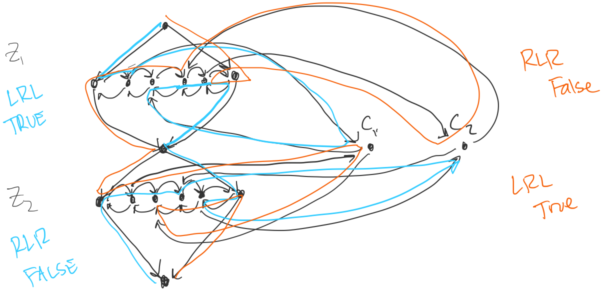
$y = \boxed{\begin{array}{l} z_1 = \text{TRUE} \\ z_2 = \text{FALSE} \end{array}}$

(also $z_1 = \text{False}$, $z_2 = \text{True}$ works)

3SAT ∈ NP b/c given an assignment, can easily check if satisfies all clauses

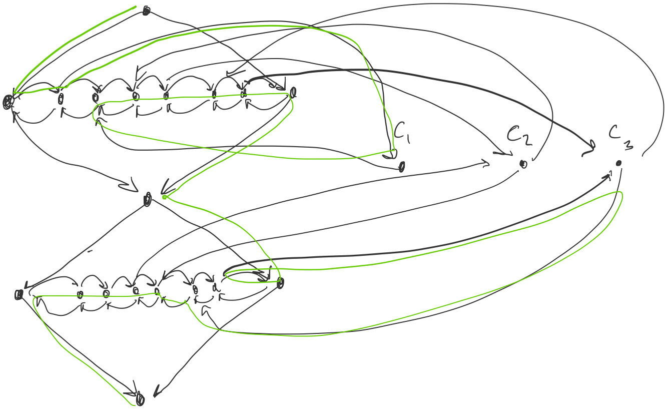
$M(x,y)$: checks if y satisfies each clause of x

$x = (z_1 \vee z_2) \wedge (\neg z_1 \vee \neg z_2)$


z_1
LRL
TRUE
C_1
C_2
RLR
False
LRL
True
z_2
RLR
FALSE

ex: $x = (z_1) \wedge (\neg z_1 \vee z_2) \wedge (\neg z_1 \vee \neg z_2)$

3SAT(x) = No (no satisfying assignment)



2. Runtime: $|x|$ (size of 3SAT input) $= O(\overset{\#\text{ clauses}}{m}) = O(\overset{\#\text{variables}}{n}{}^3)$

# of vertices is $O(nm) = O(n^4)$

at most $O(n^8)$ edges $= \binom{\#\text{ vertices}}{2}$

$O(|x|^2)$ runtime to put in all edges $\boxed{\text{Polynomial!}}$

$(n^4)^2 = n^8$

3. a) If HamPath(x') = Yes, 3SAT(x) = Yes

b) If 3SAT(x) = Yes, HamPath(x') = Yes

a) If HamPath(x') = Yes, $\exists$ a path!

→ That path must go LRL or RLR through each gadget } assign $z_i = \{$ True / False based on LRL / RLR

→ It must hit each c_i vertex once.

- Must hit c_j from a gadget z_i and return to z_i (otherwise would skip vertices)
- Then that $z_i / \bar{z}_i$ causes c_j to be satisfied.

$\underbrace{\hspace{6cm}}$

get a satisfying assignment → 3SAT(x) is Yes

b) If 3SAT(x) = Yes, there is a $\underbrace{\text{satisfying assignment}}$

- z_i = True or False
- Clause j has has at least one satisfying literal (Pick 1!)

If z_i = True ⇒ RLR

z_i = False ⇒ LRL

Pick $z_i / \bar{z}_i$ that satisfies clause j and jog out to c_j from gadget z_i.

□ QED

I would not ask you to do this from scratch!

Once you prove Π is NP-Hard, can use in place of 3-SAT:

Theorem: • If $\overset{\text{NP-Hard}}{\Pi} \le_p Q$, then $Q \in$ NP-Hard:

$\begin{matrix} A \\ \downarrow \\ B \end{matrix}$ = There is a polynomial reduction from A to B

2.4. The Web of Reductions 51

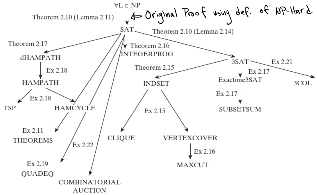

∀L ∈ NP
Theorem 2.10 (Lemma 2.11)
Original Proof using def. of NP-Hard
SAT
Theorem 2.10 (Lemma 2.14)
Theorem 2.17
Theorem 2.16
dHAMPATH
INTEGERPROG
3SAT
Ex 2.21
Ex 2.18
Theorem 2.15
Ex 2.17
HAMPATH
INDSET
Exactone3SAT
3COL
Ex 2.18
Ex 2.17
TSP
HAMCYCLE
Ex 2.15
SUBSETSUM
Ex 2.11
THEOREMS
CLIQUE
VERTEXCOVER
Ex 2.22
Ex 2.16
Ex 2.19
MAXCUT
QUADEQ
COMBINATORIAL AUCTION

Figure 2.4. Web of reductions between the **NP**-completeness problems described in this chapter and the exercises. Thousands more are known.

(From Arora + Barak)