

Lecture 34: NP-Completeness

Agenda:

- SAT and Cook's theorem
- Class NP -c
- NP -completeness proof steps
- Example proof of NP -completeness:
 $SAT \leq_p 3\text{-CNF-SAT}$

Reading:

- Textbook pages 995 – 1021

SAT — formula satisfiability and Cook's Theorem:

- Boolean formula (recursive definition):
 - Boolean variable x_i (takes value *TRUE* or *FALSE*)
 - suppose f and g are Boolean formulae
 - $\neg f$ is a Boolean formula
 - $f \vee g$ is a Boolean formula
 - $f \wedge g$ is a Boolean formula
 - $f \rightarrow g$ is a Boolean formula
 - $f \leftrightarrow g$ is a Boolean formula
- A Boolean formula is satisfiable if there is an assignment on the Boolean variables such that the formula evaluates *TRUE*
- Example: $((x_1 \rightarrow x_2) \vee \neg((\neg x_1 \leftrightarrow x_3) \vee x_4)) \wedge \neg x_2$

It is satisfiable: there is a truth assignment $x_1 = F$, $x_2 = F$, $x_3 = T$, $x_4 = T$

- SAT problem:
Instance: a Boolean formula
Query: is the formula satisfiable?

Cook's Theorem SAT is *NP*-complete.

Proof not required.

SAT variants:

- Conjunctive normal form satisfiability (CNF-SAT):
 - literal: x_i (a Boolean variable) or $\neg x_i$ (negation of x_i)
 - Boolean CNF formula: $C_1 \wedge C_2 \wedge \dots \wedge C_m$, where C_j is called a clause
 C_j is the OR of one or more literals
 - example:
 $(\neg x_3 \vee x_7) \wedge (x_1) \wedge (\neg x_2 \vee x_3 \vee \neg x_5 \vee x_6) \wedge (x_2 \vee x_3 \vee x_4)$
 - CNF-SAT:
Instance: a Boolean CNF formula
Query: is the formula satisfiable?

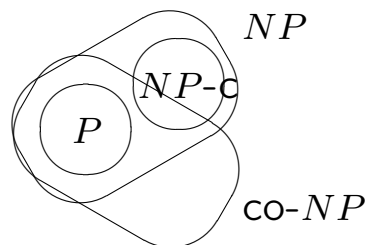
Note: CNF-SAT is a subproblem of SAT

- 3-CNF-SAT:
 - CNF formula
 - every clause contains exactly 3 distinct literals
 - example:
 $(\neg x_3 \vee x_4 \vee x_7) \wedge (x_1 \vee x_4 \vee x_6) \wedge (\neg x_2 \vee x_3 \vee x_6) \wedge (x_2 \vee x_3 \vee x_5)$
 - 3-CNF-SAT:
Instance: a Boolean 3-CNF formula
Query: is the formula satisfiable?

Note: 3-CNF-SAT is a subproblem of CNF-SAT

Class NP -c:

- Class NP -c: all NP -complete problems
 - a subclass of NP
 - NP -complete problem:
 1. is in NP
 2. every problem in NP is reducible to it
 - Cook's theorem implies that NP -c is NOT empty
- The relationships among 4 classes (most likely):



- Notes:
 1. if any NP -complete problem is in P , then $P = NP$ — unlikely
 2. least likely: $P = NP = co-NP$
 3. most likely: P , NP , $co-NP$ all different
 4. big open CS problem: is it $P = NP$ or not?

Don't try to answer at this moment!

Karp's consequence of Cook's theorem:

- Cook showed that SAT is in NP -c
- Soon after, Karp showed that 21 other problems in NP -c
- How did Karp do this so quickly?
Using the transitivity of reduction and SAT as the base NP -complete problem:
 1. SAT reduces to 3-CNF-SAT
 2. 3-CNF-SAT reduces to 3-Coloring
 3. 3-Coloring reduces to k -Coloring (fixed $k \geq 3$)
 4. SAT reduces to k -Clique
 5. k -Clique reduces to k -Independent Set
 6. ...
- Now, there are thousands of problems in NP -c
- How to prove the NP -completeness (recall):
 1. Prove that $\Pi \in NP$
 2. Look for a NP -complete problem Π'
 3. Prove that Π' reduces to Π
- We will show:
SAT $\leq_p$ 3-CNF-SAT (today)
CNF-SAT $\leq_p$ 3-CNF-SAT
3-CNF-SAT $\leq_p$ 3-Coloring
3-Coloring $\leq_p$ 4-Coloring $\leq_p$ k -Coloring ($k \geq 5$)

Proof of “SAT $\leq_p$ 3-CNF-SAT”:

- An instance I of SAT: a Boolean formula f
- Construct an instance J of 3-CNF-SAT out of I : a Boolean 3-CNF formula g
 1. construction takes polynomial time in the length of f , which can be measured by the number of Boolean operations in f
 2. f is satisfiable iff g is satisfiable

- Construction details:

- create a distinct variable for every Boolean operation in f , and then re-write the formula

for example: $f(x_1, x_2, x_3) = \neg x_1 \vee (x_2 \leftrightarrow x_3)$

create y_1 for $\neg$; y_2 for $\vee$; and y_3 for $\leftrightarrow$. Then we will have an equivalent formula

$$f'(x_1, x_2, x_3, y_1, y_2, y_3) = \begin{aligned} & y_2 \\ & \wedge (y_2 \leftrightarrow (y_1 \vee y_3)) \\ & \wedge (y_1 \leftrightarrow (\neg x_1)) \\ & \wedge (y_3 \leftrightarrow (x_2 \leftrightarrow x_3)) \end{aligned}$$

The new formula f' is in conjunctive form, in which each term involves at most 3 literals.

construction time so far:

$\Theta(\# \text{ of Boolean operations in } f)$

Construction details (cont'd):

- draw a truth table for every term and write down the CNF formula

for example: for term $y_3 \leftrightarrow (x_2 \leftrightarrow x_3)$

y_3	x_2	x_3	$y_3 \leftrightarrow (x_2 \leftrightarrow x_3)$
T	T	T	T
T	T	F	F
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	T
F	F	T	T
F	F	F	F

construction time constant (since only 3 literals)

- write down an equivalent disjunctive normal form (DNF) formula for negation of the term

for example: for term $y_3 \leftrightarrow (x_2 \leftrightarrow x_3)$

$$\neg(y_3 \leftrightarrow (x_2 \leftrightarrow x_3)) = \begin{aligned} & (y_3 \wedge x_2 \wedge \neg x_3) \\ & \vee (y_3 \wedge \neg x_2 \wedge x_3) \\ & \vee (\neg y_3 \wedge x_2 \wedge x_3) \\ & \vee (\neg y_3 \wedge \neg x_2 \wedge \neg x_3) \end{aligned}$$

construction time constant

- the term itself is the negation of the negation (DeMorgan Laws)

for example: for term $y_3 \leftrightarrow (x_2 \leftrightarrow x_3)$

$$\begin{aligned} y_3 \leftrightarrow (x_2 \leftrightarrow x_3) &= \neg(\neg(y_3 \leftrightarrow (x_2 \leftrightarrow x_3))) \\ &= \neg(\neg y_3 \vee \neg x_2 \vee x_3) \\ &= \neg(\neg y_3 \vee x_2 \vee \neg x_3) \\ &= \neg(y_3 \vee \neg x_2 \vee \neg x_3) \\ &= \neg(y_3 \vee x_2 \vee x_3) \end{aligned}$$

construction time constant

Proof of “ $\text{SAT} \leq_p \text{3-CNF-SAT}$ ” — conclusion:

- An instance I of SAT: a Boolean formula f
- Construct an instance J of 3-CNF-SAT out of I : a Boolean 3-CNF formula g
 1. construction takes linear time in the number of Boolean operations in formula f , and thus polynomial time
 2. f is satisfiable iff g is satisfiable

Proof.

notice that g is equivalent to f in the sense that those y literals can be assigned values according to the values of x literals.

therefore, if f is satisfiable, then g is satisfiable; on the other hand, if g is satisfiable, then the x literals inherit the values in the truth assignment is an assignment on which f evaluates to TRUE.

in the other words, f is satisfiable iff g is satisfiable.

- Conclusion:
 - we just showed $\text{SAT} \leq_p \text{3-CNF-SAT}$
 - $\text{3-CNF-SAT} \in NP$
 - therefore, 3-CNF-SAT is NP -complete

Lecture 34: NP-Completeness

Have you understood the lecture contents?

well	ok	not-at-all	topic
☐	☐	☐	SAT and Cook's theorem
☐	☐	☐	Class NP -c, properties
☐	☐	☐	How to prove the NP -completeness
☐	☐	☐	$SAT \leq_p 3\text{-CNF-SAT}$