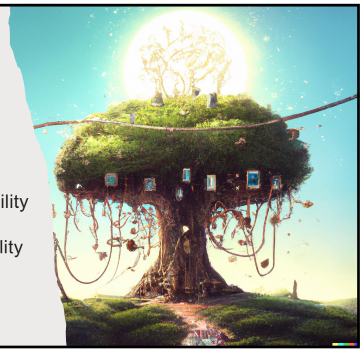
CS21
Decidability and Tractability

Lecture 20
February 24, 2025



1

Outline

- 3-SAT is NP-complete
- NP-complete problems: independent set, vertex cover, clique…
- NP-complete problems: Hamilton path and cycle, Traveling Salesperson Problem
- NP-complete problems: Subset Sum
- NP-complete problems: NAE-3-SAT, max cut

2

CIRCUIT-SAT is NP-complete

Theorem: CIRCUIT-SAT is NP-complete

CIRCUIT-SAT = {C : C is a Boolean circuit for which there exists a satisfying truth assignment}

Proof:

- Part 1: need to show CIRCUIT-SAT $\in$ NP.
  - can express CIRCUIT-SAT as:

CIRCUIT-SAT = {C : C is a Boolean circuit for which $\exists$x such that (C, x) $\in$ R}

R = {(C, x) : C is a Boolean circuit and C(x) = 1}

3

CIRCUIT-SAT is NP-complete

CIRCUIT-SAT = {C : C is a Boolean circuit for which there exists a satisfying truth assignment}

Proof:

- Part 2: for each language A $\in$ NP, need to give poly-time reduction from A to CIRCUIT-SAT
- for a given language A $\in$ NP, we know

$A = \{x \mid \exists\, y, |y| \le |x|^k, (x, y) \in R\}$

and there is a (deterministic) TM M_R that decides R in time $g(n) \le n^c$ for some c.

4

CIRCUIT-SAT is NP-complete

- Tableau (configurations written in an array) for machine M_R on input w = (x, y):

w_1/q_s	w_2	…	w_n	…	_
w_1	w_2/q_1	…	w_n	…	_
w_1/q_1	a	…	w_n	…	_
⋮					⋮
_/q_a	_	…	_	…	_

- height = time taken = $|w|^c$
- width = space used $\le |w|^c$

5

CIRCUIT-SAT is NP-complete

- Important observation: contents of cell in tableau determined by 3 others above it:

a/q_1	b	a
	b/q_1	

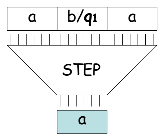
a	b/q_1	a
	a	

a	b	a
	b	

6

CIRCUIT-SAT is NP-complete

- Can build Boolean circuit STEP
  - input (binary encoding of) 3 cells
  - output (binary encoding of) 1 cell

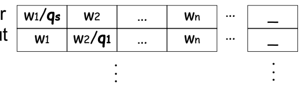


- each output bit is some function of inputs
- can build circuit for each
- size is independent of size of tableau

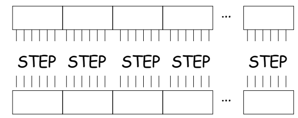
7

CIRCUIT-SAT is NP-complete

Tableau for M_R on input $w = (x, y)$

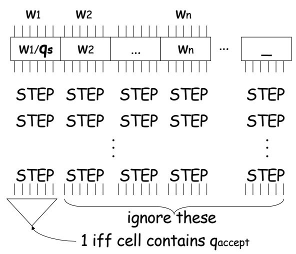


- $|w|^c$ copies of STEP compute row i from i-1



8

CIRCUIT-SAT is NP-complete



This circuit $C_{M_R, w}$ has inputs $w_1w_2\ldots w_n$

and $C(w) = 1$ iff M_R accepts input w.

Size = $O(|w|^{2c})$

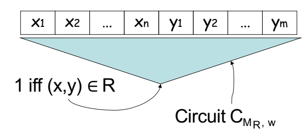
9

CIRCUIT-SAT is NP-complete

- recall: we are reducing language A:

  $A = \{ x \mid \exists y, |y| \leq |x|^k, (x, y) \in R \}$

  to CIRCUIT-SAT.
- f(x) produces the following circuit:



- hardwire input x
- leave y as variables

10

CIRCUIT-SAT is NP-complete

- is f(x) poly-time computable?
  - hardcode M_R, k and c
  - circuit has size $O(|w|^{2c})$; $|w| = |(x,y)| \leq n + n^k$
  - each component easy to describe efficiently from description of M_R
- YES maps to YES?
  - $x \in A \Rightarrow \exists y$, M_R accepts $(x, y) \Rightarrow f(x) \in$ CIRCUIT-SAT
- NO maps to NO?
  - $x \notin A \Rightarrow \forall y$, M_R rejects $(x, y) \Rightarrow f(x) \notin$ CIRCUIT-SAT

11

3SAT is NP-complete

Theorem: 3SAT is NP-complete

3SAT = {φ : φ is a 3-CNF formula for which there exists a satisfying truth assignment}

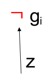
Proof:

- Part 1: need to show 3-SAT ∈ NP
  - already done
- Part 2: need to show 3-SAT is NP-hard
  - we will give a poly-time reduction from CIRCUIT-SAT to 3-SAT

12

3SAT is NP-complete

- given a circuit C
  - variables $x_1, x_2, \ldots, x_n$
  - AND ($\wedge$), OR ($\vee$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
- reduction f(C) produces these clauses for φ on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:



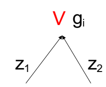
- $(g_i \vee z)$
- $(\neg z \vee \neg g_i)$

$(z \Leftrightarrow \neg g_i)$

13

3SAT is NP-complete

- given a circuit C
  - variables $x_1, x_2, \ldots, x_n$
  - AND ($\wedge$), OR ($\vee$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
- reduction f(C) produces these clauses for φ on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:



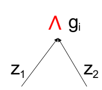
- $(\neg z_1 \vee g_i)$
- $(\neg z_2 \vee g_i)$
- $(\neg g_i \vee z_1 \vee z_2)$

$(z_1 \vee z_2 \Leftrightarrow g_i)$

14

3SAT is NP-complete

- given a circuit C
  - variables $x_1, x_2, \ldots, x_n$
  - AND ($\wedge$), OR ($\vee$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
- reduction f(C) produces these clauses for φ on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:



- $(\neg g_i \vee z_1)$
- $(\neg g_i \vee z_2)$
- $(\neg z_1 \vee \neg z_2 \vee g_i)$

$(z_1 \wedge z_2 \Leftrightarrow g_i)$

15

3SAT is NP-complete

- finally, reduction f(C) produces single clause (g_m) where g_m is the output gate.
- f(C) computable in poly-time?
  - yes, simple transformation
- YES maps to YES?
  - if C(x) = 1, then assigning x-values to x-variables of φ and gate values of C when evaluating x to the g-variables of φ gives satsifying assignment.

16

3SAT is NP-complete

- NO maps to NO?
  - show that φ satisfiable implies C satisfiable
  - satisfying assignment to φ assigns values to x-variables and g-variables
  - output gate g_m must be assigned 1
  - every other gate must be assigned value it would take given values of its inputs.
  - the assignment to the x-variables must be a satisfying assignment for C.

17

Search vs. Decision

- Definition: given a graph G = (V, E), an independent set in G is a subset $V' \subseteq V$ such that for all $u,w \in V'$ $(u,w) \notin E$
- A problem:

  given G, find the largest independent set
- This is called a search problem
  - searching for *optimal* object of some type
  - comes up frequently

18

Search vs. Decision

- We want to talk about languages (or decision problems)
- Most search problems have a natural, related decision problem by adding a bound "k"; for example:
  - search problem: given G, find the largest independent set
  - decision problem: given (G, k), is there an independent set of size *at least* k

Ind. Set is NP-complete

Theorem: the following language is NP-complete:

IS = {(G, k) : G has an IS of size $\geq$ k}.

- Proof:
  - Part 1: IS $\in$ NP. Proof?
  - Part 2: IS is NP-hard.
    - reduce from 3-SAT

Ind. Set is NP-complete

- We are reducing from the language:

3SAT = { φ : φ is a 3-CNF formula that has a satisfying assignment }

to the language:

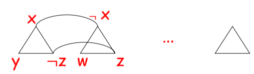
IS = {(G, k) : G has an IS of size $\geq$ k}.

Ind. Set is NP-complete

The reduction f: given

$\varphi = (x \vee y \vee \neg z) \wedge (\neg x \vee w \vee z) \wedge \ldots \wedge (\ldots)$

we produce graph G_φ:

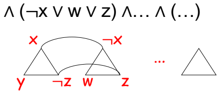


- one triangle for each of m clauses
- edge between every pair of contradictory literals
- set k = m

Ind. Set is NP-complete

$\varphi = (x \vee y \vee \neg z) \wedge (\neg x \vee w \vee z) \wedge \ldots \wedge (\ldots)$

$f(\varphi)$ = (G, # clauses)

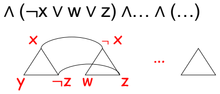


- Is f poly-time computable?
- YES maps to YES?
  - 1 true literal per clause in satisfying assign. A
  - choose corresponding vertices (1 per triangle)
  - IS, since no contradictory literals in A

Ind. Set is NP-complete

$\varphi = (x \vee y \vee \neg z) \wedge (\neg x \vee w \vee z) \wedge \ldots \wedge (\ldots)$

$f(\varphi)$ = (G, # clauses)



- NO maps to NO?
  - IS can have at most 1 vertex per triangle
  - IS of size $\geq$ # clauses must have exactly 1 per
  - since IS, no contradictory vertices
  - can produce satisfying assignment by setting these literals to true

Vertex cover

- Definition: given a graph G = (V, E), a vertex cover in G is a subset V' ⊆ V such that for all (u,w) ∈ E, u ∈ V' or w ∈ V'
- A search problem:
  given G, find the smallest vertex cover
- corresponding language (decision problem):
  VC = {(G, k) : G has a VC of size ≤ k}.

Vertex Cover is NP-complete

Theorem: the following language is NP-complete:
VC = {(G, k) : G has a VC of size ≤ k}.

- Proof:
  - Part 1: VC ∈ NP. Proof?
  - Part 2: VC is NP-hard.
    - reduce from?

Vertex Cover is NP-complete

- We are reducing from the language:

IS = {(G, k) : G has an IS of size ≥ k}

to the language:

VC = {(G, k) : G has a VC of size ≤ k}.

Vertex Cover is NP-complete

- How are IS, VC related?
- Given a graph G = (V, E) with n nodes
  - if V' ⊆ V is an independent set of size k
  - then V-V' is a vertex cover of size n - k
- Proof:
  - suppose not. Then there is some edge with neither endpoint in V-V'. But then both endpoints are in V'. contradiction.

Vertex Cover is NP-complete

- How are IS, VC related?
- Given a graph G = (V, E) with n nodes
  - if V' ⊆ V is a vertex cover of size k
  - then V-V' is an independent set of size n - k
- Proof:
  - suppose not. Then there is some edge with both endpoints in V-V'. But then neither endpoint is in V'. contradiction.

Vertex Cover is NP-complete

The reduction:
- given an instance of IS: (G, k) f produces the pair (G, n-k)

- f poly-time computable?
- YES maps to YES?
  - IS of size ≥ k in G ⇒ VC of size ≤ n-k in G
- NO maps to NO?
  - VC of size ≤ n-k in G ⇒ IS of size ≥ k in G