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

Cook-Levin Theorem

- Gateway to proving lots of natural, important problems NP-complete is:

  **Theorem** (Cook, Levin): 3SAT is NP-complete.

  **Recall:** 3SAT = \{\phi : \phi \text{ is a CNF formula with 3 literals per clause for which there exists a satisfying truth assignment}\}

CIRCUIT-SAT is NP-complete

**Theorem:** CIRCUIT-SAT is NP-complete

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

3SAT is NP-complete

**Theorem:** 3SAT is NP-complete

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

**Proof:**

- Part 1: need to show 3-SAT \(\in\) 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
3SAT is NP-complete

– given a circuit C
  • variables $x_1, x_2, \ldots, x_n$
  • AND ($\land$), OR ($\lor$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
– reduction $f(C)$ produces these clauses for $\varphi$
on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:

\[ \neg g_i \rightarrow (g_i \lor z) \times (\neg z \lor \neg g_i) \]

\[ (z \equiv \neg g_i) \]

3SAT is NP-complete

– given a circuit C
  • variables $x_1, x_2, \ldots, x_n$
  • AND ($\land$), OR ($\lor$), NOT ($\neg$) gates $g_1, g_2, \ldots, g_m$
– reduction $f(C)$ produces these clauses for $\varphi$
on variables $x_1, x_2, \ldots, x_n, g_1, g_2, \ldots, g_m$:

\[ \neg g_i \rightarrow (g_i \lor z) \times (\neg z \lor \neg g_i) \]

\[ (z \equiv \neg g_i) \]

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

NO maps to NO?

• show that $\varphi$ satisfiable implies C satisfiable
• satisfying assignment to $\varphi$ 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.
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:
\[ IS = \{(G, k) : G \text{ has an IS of size } \geq k \} . \]

Proof:
– Part 1: IS ∈ 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:
\[ IS = \{(G, k) : G \text{ has an IS of size } \geq k \} . \]

Ind. Set is NP-complete
The reduction f: given
\[ φ = (x \lor y \lor \neg z) \land (\neg x \lor w \lor z) \land \ldots \land (\ldots) \]
we produce graph \( G_φ \):

\[ x \rightarrow y \rightarrow \neg z \rightarrow w \rightarrow \neg z \rightarrow \ldots \rightarrow \triangle \]

• one triangle for each of m clauses
• edge between every pair of contradictory literals
• set \( k = m \)

Ind. Set is NP-complete
\[ φ = (x \lor y \lor \neg z) \land (\neg x \lor w \lor z) \land \ldots \land (\ldots) \]
f(φ) = \( (G, \# \text{ clauses}) \)

• 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
\[ φ = (x \lor y \lor \neg z) \land (\neg x \lor w \lor z) \land \ldots \land (\ldots) \]
f(φ) = \( (G, \# \text{ clauses}) \)

• NO maps to NO?
  – IS can have at most 1 vertex per triangle
  – IS of size \( \geq \# \text{ clauses} \) 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 \( V' \subseteq V \) such that for all \((u,w) \in E\), \( u \in V' \) or \( w \in V' \).
- A search problem: given \( G \), find the smallest vertex cover.
- A corresponding language (decision problem):
  \[ VC = \{(G, k) : G \text{ has a VC of size } \leq k\}. \]

Vertex Cover is NP-complete

Theorem: the following language is NP-complete:
\[ VC = \{(G, k) : G \text{ has a VC of size } \leq k\}. \]

- Proof:
  - Part 1: \( VC \in \text{NP}. \) Proof?
  - Part 2: \( VC \) is NP-hard.
    - reduce from?

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 \( \geq k \) in \( G \implies VC \text{ of size } \leq n-k \) in \( G \)
- NO maps to NO?
  - VC of size \( \leq n-k \) in \( G \implies IS \text{ of size } \geq k \) in \( G \)

- We are reducing from the language:
  \[ IS = \{(G, k) : G \text{ has an IS of size } \geq k\} \]
to the language:
\[ VC = \{(G, k) : G \text{ has a VC of size } \leq k\}. \]

- How are IS, VC related?
  - Given a graph \( G = (V, E) \) with \( n \) nodes
    - if \( V' \subseteq V \) 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.

- Given a graph \( G = (V, E) \) with \( n \) nodes
  - if \( V' \subseteq V \) 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.
Clique

- Definition: given a graph \( G = (V, E) \), a clique in \( G \) is a subset \( V' \subseteq V \) such that for all \( u, v \in V' \), \( (u, v) \in E \)
- A search problem:
  given \( G \), find the largest clique
- Corresponding language (decision problem):
  \( \text{CLIQUE} = \{ (G, k) : G \text{ has a clique of size } \geq k \} \)

Clique is NP-complete

**Theorem:** the following language is NP-complete:
\( \text{CLIQUE} = \{ (G, k) : G \text{ has a clique of size } \geq k \} \)

- Proof:
  - Part 1: CLIQUE \( \in \) NP. Proof?
  - Part 2: CLIQUE is NP-hard.
    - reduce from?

Clique is NP-complete

- How are IS, CLIQUE related?
- Given a graph \( G = (V, E) \), define its complement \( G' = (V, E' = \{(u,v) : (u,v) \notin E\}) \)
  - if \( V' \subseteq V \) is an independent set in \( G \) of size \( k \)
  - then \( V' \) is a clique in \( G' \) of size \( k \)
- Proof:
  - Every pair of vertices \( u, v \in V' \) has an edge between them in \( G' \). Therefore they have an edge between them in \( G \).

Clique is NP-complete

The reduction:
  - given an instance of IS: \( (G, k) \) f produces the pair \( (G', k) \)
- \( f \) poly-time computable?
- YES maps to YES?
  - IS of size \( \geq k \) in \( G \) \( \Rightarrow \) CLIQUE of size \( \geq k \) in \( G' \)
- NO maps to NO?
  - CLIQUE of size \( \geq k \) in \( G' \) \( \Rightarrow \) IS of size \( \geq k \) in \( G \)