Introduction to the Theory NP-Completeness

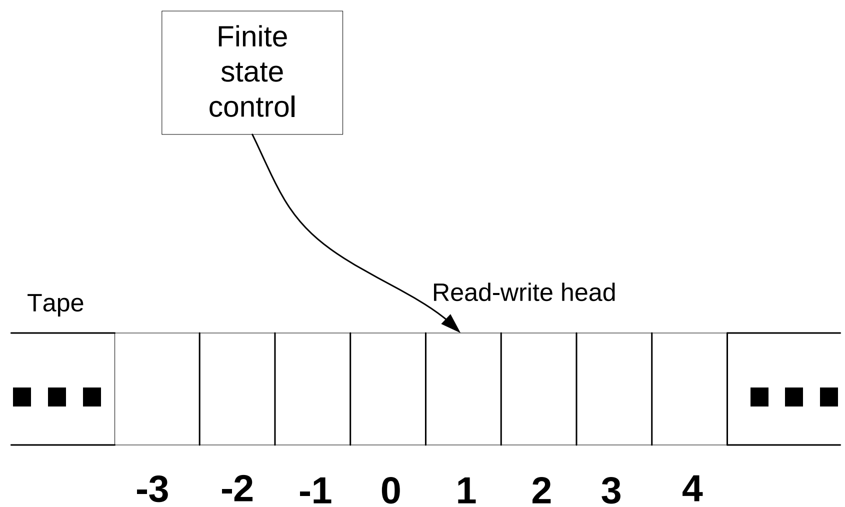
(Reference book: Computers and Intractability: A Guide to the Theory of NP-Completeness by Michael Gary and David Johnson)

- P: The set of problems that can be solved in polynomial time in a deterministic Turing machine

Introduction to the Theory NP-Completeness

(Reference book: Computers and Intractability: A Guide to the Theory of NP-Completeness by Michael Gary and David Johnson

- P: The set of problems that can be solved in polynomial time in a deterministic Turing machine

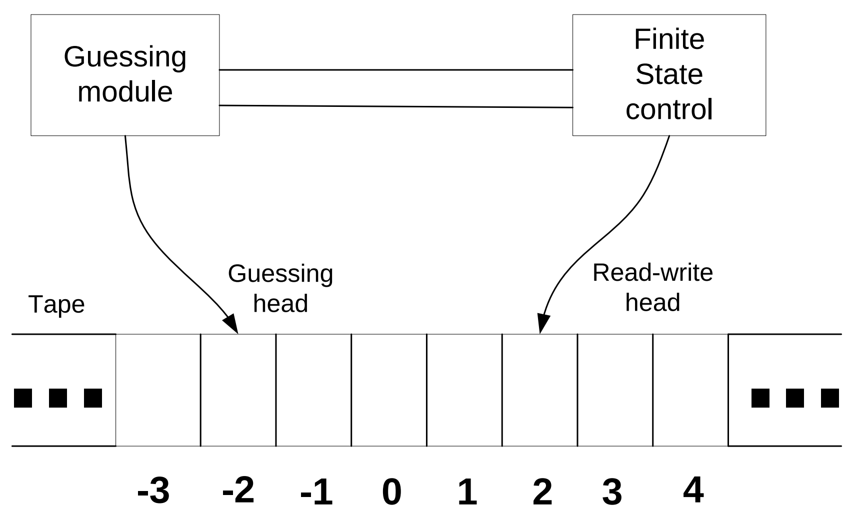


Introduction to the Theory NP-Completeness

(Reference book: Computers and Intractability: A Guide to the Theory of NP-Completeness by Michael Gary and David Johnson

- P: The set of problems that can be solved in polynomial time in a deterministic Turing machine

- NP: The set of problems that can be solved in polynomial time in a non-deterministic Turing machine



Introduction to the Theory NP-Completeness

(Reference book: Computers and Intractability: A Guide to the Theory of NP-Completeness by Michael Gary and David Johnson

- P: The set of problems that can be solved in polynomial time in a deterministic Turing machine
- NP: The set of problems that can be solved in polynomial time in a non-deterministic Turing machine
- NP-Complete: A problem X is NP-Complete if it is a member of the set NP and every problem in NP can be transformed to X in polynomial time
- Second Definition: If a problem X is NP-Complete and X can be transformed in polynomial time to another problem Y, then Y is also an NP-Complete problem.

NP-Completeness

- Non-deterministic Turing machine: A NDTM has a "guessing module" and a "checking module". The guessing module comes up with a solution and the checking module verifies if the guessed solution is correct.

- Polynomial Time Nondeterministic Algorithm is basically a definitional device for capturing the notion of polynomial time verifiability, rather than a realistic method for solving decision problems.

NP-Completeness

- Let $U = \{u_1, u_2, \ldots, u_m\}$ be a set of Boolean variables. A truth assignment for U is a function $t: U \rightarrow \{T, F\}$. If $t(u) = T$ we say that u is "true" under t; if $t(u) = F$ we say that u is "false". If u is a variable in U, then u and $\bar{u}$ are literals over U. The literal u is true under t if and only if the variable u is true under t; the literal $\bar{u}$ is true, if and only if the variable u is false under t.

NP-Completeness

- A clause c is a set of literals over U. It represents the disjunction of those literals and is satisfied by a truth assignment if and only if at least one member is true under that assignment.

- For example, $c_1 = \{u_1, \overline{u}_3, u_8\}$. This will be satisfied by t, unless $t(u_1) = F$, $t(u_3) = T$ and $t(u_8) = F$

- A collection C of clauses over U is satisfied if and only if there exists some truth assignment for U, that simultaneously satisfies all clauses in C.

NP-Completeness

- SATISFIABILITY Problem
  - Instance: A set U of variables and a collection C of clauses over U.
  - Question: Is there a satisfying truth assignment?
- SATISFIABILITY is NP-Complete (Cook’s Theorem)
- SATISFIABILITY is the first NP-Complete problem
- How do you prove the first NP-Complete problem?

Basic NP-Complete Problems

3 - SATISFIABILITY (3SAT)

INSTANCE: Collection $C=\{c_1, c_2, \dots, c_m\}$ of clauses on a finite set U of variables such that $|c_i|=3$ for $1 \le i \le m$

QUESTION: Is there a truth assignment for U that satisfies all the clauses in C?

3 – DIMENSIONAL MATCHING (3DM)

INSTANCE: A set $M \subseteq W \times X \times Y$, where W, X and Y are disjoint sets having the same number q of elements.

QUESTION: Does M contain a matching, that is a subset $M' \subseteq M$ such that $|M'| = q$ and no two elements of M' agree in any coordinate?

VERTEX COVER (VC)

INSTANCE: A graph G=(V,E) and a positive integer $K \le |V|$

QUESTION: Is there a vertex cover of size K or less for G, that is, a subset $V' \subseteq V$ such that $|V'| \le k$ and, for each edge $\{u,v\} \in E$, at least one of u and v belongs to V' ?

CLIQUE

INSTANCE: A graph G = (V, E) and a positive integer $J \le |V|$

QUESTION: Does G contain a clique of size J or more, that is, a subset $V' \subseteq V$ such that $|V'| \ge J$ and every two vertices in V' are joined by an edge in E?

HAMILTONIAN CIRCUIT (HC)

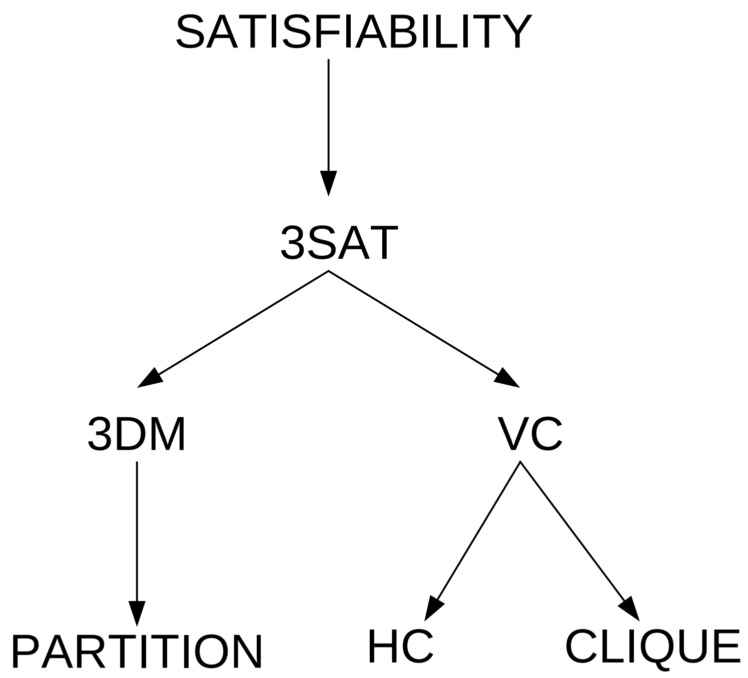
INSTANCE: A graph G = (V, E)

QUESTION: Does G contain a Hamiltonian circuit, that is, an ordering $< v_1, v_2, \ldots, v_n >$ of the vertices of G, where $n = |V|$, such that $\{v_n, v_1\} \in E$ and $\{v_i, v_{i+1}\} \in E$ for all i, $1 \le i < n$?

PARTITION

INSTANCE: A finite set A and a "size" $s(a) \in Z^+$ for each $a \in A$

QUESTION: Is there a subset $A' \subseteq A$ such that $\sum_{a \in A'} s(a) = \sum_{a \in A - A'} s(a)$?



Sequence of transformations used to prove NP-Completeness

- 3-SAT is NP-Complete
- Proof by transformation from the SATISFIABILITY Problem
- Two steps are involved
  - Step 1: From an instance of the SAT problem I_{SAT} generate an instance of the 3-SAT problem $I_{3\text{-}SAT}$
  - Step2: Prove that $I_{SAT} \in Y_{SAT}$ if and only if $I_{3\text{-}SAT} \in Y_{3\text{-}SAT}$
- Generation of an instance of 3-SAT from an instance of SAT
  - An instance of SAT is specified by
    - A set of variables $U = \{u_1, \ldots, u_n\}$
    - A set of clauses $C = \{c_1, \ldots, c_m\}$

- I_{SAT}: $U = \{u_1, \ldots, u_n\}$ $C = \{c_1, \ldots, c_m\}$

- We will construct a collection C′ of three literal clauses on a set U′ of variables such that C′ is satisfiable if and only if C is satisfiable
- The construction of C′ will merely replace each individual clause c_j є C by an "equivalent" collection C'_j of three literal clauses, based on the original variables U and some additional variables U'_j whose use will be limited to clauses on C'_j. The variables and the clauses in $I_{3\text{-}SAT}$ will be

$$U' = U \cup \left(\bigcup_{j=1}^{m} U'_j \right)$$

$$C' = \bigcup_{j=1}^{m} C'_j$$

- Let c_j be given by $\{z_1, z_2, \ldots, z_k\}$ where z_i's are all literals derived from the variables in U. The way in which C'_j and U'_j are formed depends on the value of k

k = 1

$$U'_j = \{y_j^1, y_j^2\}$$

$$C'_j = \{\{z_1, y_j^1, y_j^2\}, \{z_1, y_j^1, \overline{y}_j^2\}, \{z_1, \overline{y}_j^1, y_j^2\}, \{z_1, \overline{y}_j^1, \overline{y}_j^2\}\}$$

k = 2

$$U'_j = \{y_j^1\} \quad C'_j = \{\{z_1, z_2, y_j^1\}, \{z_1, z_2, \overline{y}_j^1\}\}$$

k = 3

$$U'_j = \phi \quad C'_j = \{\{c_j\}\}$$

k > 3

$$U'_j = \{y_j^i : 1 \le i \le k-3\}$$

$$C'_j = \{\{z_1, z_2, y_j^1\}\} \cup \{\{\overline{y}_j^i, z_{i+2}, y_j^{i+1}\} : 1 \le i \le k-4\} \cup \{\{\overline{y}_j^{k-3}, z_{k-1}, z_k\}\}$$

- We need to show that

$$I_{SAT} \in Y_{SAT} \Leftrightarrow I_{3\text{-}SAT} \in Y_{3\text{-}SAT}$$

Step 1: $I_{SAT} \in Y_{SAT} \Leftarrow I_{3\text{-}SAT} \in Y_{3\text{-}SAT}$

Step 2: $I_{SAT} \in Y_{SAT} \Rightarrow I_{3\text{-}SAT} \in Y_{3\text{-}SAT}$

- If t is a satisfying truth assignment of the set of clauses C, we need to show how t can be extended to t': $U' \rightarrow \{T, F\}$ satisfying C'.

- Since the variables in U'- U are partitioned into sets U'_j and since the variables in each U'_j occur only in clauses belonging to C'_j, we need only to show how t can be extended to the sets U'_j one at a time, and in each case we need only to verify that all the clauses in the corresponding C'_j, are satisfied

When k = 1, 2 or 3, then the clauses in C'_j are already satisfied by t and we can arbitrarily extend t to U'_j

When k > 3, we do the following:

Since C is a satisfying truth assignment, there must be at least one literal that is set true by t

l **= least integer such that the literal** z_l **is set true under t**

$$l = 1 \, or \, 2 \qquad t'(y_j^i) = F, \quad 1 \le i \le k-3$$

$$l = k-1 \, or \, k \qquad t'(y_j^i) = T, \quad 1 \le i \le k-3$$

$$otherwise \qquad t'(y_j^i) = T, \quad 1 \le i \le l-2$$

$$and \qquad t'(y_j^i) = F, \quad l-1 \le i \le k-3$$

- Vertex Cover (VC) Problem is NP-Complete

- Proof by transformation from the 3-SAT Problem

- Two steps are involved
  - Step 1: From an instance of the 3-SAT problem $I_{3\text{-SAT}}$ generate an instance of the VC problem I_{VC}
  - Step2: Prove that $I_{3\text{-SAT}} \in Y_{3\text{-SAT}}$ if and only if $I_{VC} \in Y_{VC}$

- Generation of an instance of VC from an instance of 3-SAT
  - An instance of 3-SAT is specified by
    - A set of variables $U = \{u_1, \ldots, u_n\}$
    - A set of clauses, each with three literals $C = \{c_1, \ldots, c_m\}$

3-SAT $U = \{u_1, \ldots, u_n\}$

$C = \{c_1, \ldots, c_m\}$

VC $G = (V, E),\ k$

$$\forall u_i \in U \ \Rightarrow \ \mathrm{T}_i = (V_i, E_i)$$

$$V_i = \{u_i, \overline{u}_i\},\ E_i = \{\{u_i, \overline{u}_i\}\}$$

Truth Setting Component

$$\forall c_j \in C \ \Rightarrow \ S_j = (V_j^{'}, E_j^{'})$$

Satisfaction Testing Component

$$V_j^{'} = \{a_1[j], a_2[j], a_3[j]\}$$

$$E_j^{'} = \{\{a_1[j], a_2[j]\}, \{a_2[j], a_3[j]\}, \{a_3[j], a_1[j]\}\}$$

$$E_j^{"} = \{\{a_1[j], x_j\}, \{a_2[j], y_j\}, \{a_3[j], z_j\}\}$$

Communication Edges

$$c_j = (x_j, y_j, z_j)$$

$$G = (V, E)$$

Set k = n + 2m

$$V = (\bigcup_{i=1}^{n} V_i) \cup (\bigcup_{j=1}^{m} V_j^{'}) \quad E = (\bigcup_{i=1}^{n} E_i) \cup (\bigcup_{j=1}^{m} E_j^{'}) \cup (\bigcup_{j=1}^{m} E_j^{"})$$

Example: Generation of a instance of VC from an instance of 3-SAT

$I_{3\text{-SAT}}$: $U = \{u_1, u_2, u_3, u_4\}$, $C = \{\{u_1, \bar{u}_3, \bar{u}_4\}, \{\bar{u}_1, u_2, \bar{u}_4\}\}$