

NP-Completeness

An *abstract problem* is a binary relation on a set of instances and a set of solutions

Problem	Instance	Solution
SORTING	Array of keys	π , permutation of indices that gives sorted order
SHORTEST-PATH	A graph $G=(V,E)$ and 2 vertices	The length of the shortest path between the 2 vertices
CLIQUE	A graph $G=(V,E)$	The size of the largest clique in G
SATISFIABILITY	A boolean formula $\varphi()$	A truth assignment for $\varphi()$ which satisfies it (makes it evaluate to True) or nil

NP-Completeness (cont)

An *algorithm* for an abstract problem is a mapping which is a subset of the problem (relation)

SORTING

$$A_{\text{SORTING}} : \{\text{Array of keys}\} \rightarrow \{\pi, \text{permutation of indices}\}$$

An *abstract decision problem* is an abstract problem whose solution set is {Yes,No} (or {1,0})

A decision problem is specified by giving the set of instances and a Yes/No question.

2

NP-Completeness (cont)

Decision Problem	Instance	Question
SORTING	(Array of keys, π)	Is array in order π sorted?
SHORTEST-PATH	($G=(V,E)$, s , t , k)	Is there a path of length no more than k between s and t ?
CLIQUE	($G=(V,E)$, k)	Is there a clique of size at least k in G ?
SATISFIABILITY	A boolean formula $\varphi()$	Is there a truth assignment for $\varphi()$ which satisfies it?

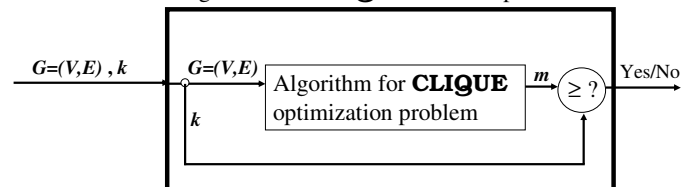
3

NP-Completeness (cont)

An abstract problem's algorithm can be used to solve the related decision problem directly.

CLIQUE

Algorithm for **CLIQUE** decision problem



4

NP-Completeness (cont)

A decision problem's algorithm can be used to solve the related abstract optimization problem using a search method on the solution space.

```
min ← -1   max ← |V|+1
while (min < max-1)
  mid ← (max-min)/2
  If (G has a clique of size ≥ mid)
    then min ← mid
    else max ← mid
endwhile
return(min)
```

5

NP-Completeness (cont)

Encodings

Complexity of a problem can vary depending on the encoding since complexity is measured as a function of the input size.

Input encoding can be artificially padded or contain the solution to make the complexity appear smaller.

A *concrete problem* has as its instance set the set of binary strings.

An algorithm solves a concrete problem in $O(T(n))$ time if it always produces the solution within time $O(T(n))$ where n is the length of the binary input string.

6

NP-Completeness (cont)

Encodings

A concrete problem is *polynomially solvable* if there is an algorithm which solves it in time $O(n^k)$ for some constant k .

$\mathbf{P} = \{\text{polynomially solvable concrete decision problems}\}$

$e(\mathbf{SORTING}), e(\mathbf{SHORTEST-PATH}) \in \mathbf{P}$

$e(\mathbf{CLIQUE}), e(\mathbf{SATISFIABILITY}) \in \mathbf{P}?$

Can different encodings of an abstract decision problem affect membership in \mathbf{P} ?

7

NP-Completeness (cont)

$e : I \rightarrow \{0,1\}^*$ an encoding of an abstract decision problem.

Two encodings e_1 and e_2 of an abstract decision problem are polynomially related if there exist functions

$$f_{12} \text{ and } f_{21} : \{0,1\}^* \rightarrow \{0,1\}^*$$

such that

\forall instance i , $f_{12}(e_1(i)) = e_2(i)$ and $f_{21}(e_2(i)) = e_1(i)$

and f_{12} and f_{21} are polynomial time computable.

Prevents padding of input, unary encodings and inclusion of solutions (unless solution can be found in polynomial time.)

8

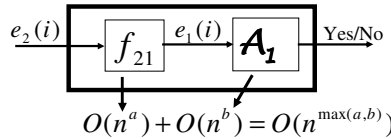
NP-Completeness (cont)

Lemma If \mathcal{Q} is an abstract decision problem with two encodings e_1 and e_2 that are polynomially related, then

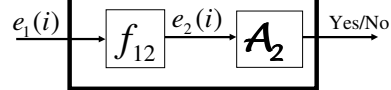
$$e_1(\mathcal{Q}) \in \mathbf{P} \Leftrightarrow e_2(\mathcal{Q}) \in \mathbf{P}$$

Proof:

$$e_1(\mathcal{Q}) \in \mathbf{P} \Rightarrow e_2(\mathcal{Q}) \in \mathbf{P}$$



$$e_2(\mathcal{Q}) \in \mathbf{P} \Rightarrow e_1(\mathcal{Q}) \in \mathbf{P}$$



QED 9

NP-Completeness (cont)

For an abstract decision problem \mathcal{Q} and an encoding $e(\)$

$$L_{\mathcal{Q}} = \{ e(i) \mid \text{the solution for } i \text{ is Yes} \}$$

Example : $L_{\text{CLIQUE}} = \{ e(G,k) \mid G \text{ has a clique of size } \geq k \}$

L_1 is polynomial time reducible to L_2 if

$\exists f : \{0,1\}^* \rightarrow \{0,1\}^*$ computable in polynomial time

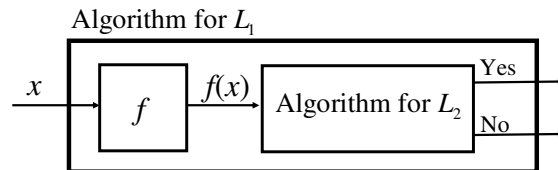
such that $x \in L_1 \Leftrightarrow f(x) \in L_2$

$$L_1 \leq_p L_2$$

10

NP-Completeness (cont)

If L_1 is polynomial time reducible to L_2



$$L_2 \in \mathbf{P} \Rightarrow L_1 \in \mathbf{P} \quad \text{"} L_1 \text{ is not harder than } L_2 \text{"}$$

$$L_1 \notin \mathbf{P} \Rightarrow L_2 \notin \mathbf{P} \quad L_1 \leq_p L_2$$

11

NP-Completeness (cont)

To show $L_1 \leq_p L_2$

Step 1: Construct $f : \{0,1\}^* \rightarrow \{0,1\}^*$ mapping instances of L_1 to L_2

Step 2: Show that $x \in L_1 \Leftrightarrow f(x) \in L_2$

Step 3: Show that $f(\)$ can be computed in polynomial time

In terms of abstract decision problems, this means an instance of \mathcal{Q}_1 can be converted to an instance of \mathcal{Q}_2 in polynomial time so that the answer (Yes or No) is the same for both instances.

12

NP-Completeness (cont)

3SAT (or 3CNF-SAT)

Instance: A boolean formula ϕ in conjunctive normal form in which each clause has exactly 3 distinct literals.

Question: Can this formula ϕ be satisfied?

(Is there an assignment of values to its variables that makes it evaluate to 1?)

$$\phi_1 = (x_1 \vee x_2 \vee \bar{x}_3)(\bar{x}_2 \vee x_3 \vee \bar{x}_4)(\bar{x}_1 \vee \bar{x}_2 \vee x_4)$$

$x_1 = x_3 = x_4 = 1 \quad x_2 = 0$

ϕ_1 is satisfiable

$$\phi_2 = (x_1 \vee x_2 \vee x_3)(\bar{x}_1 \vee x_2 \vee x_4)(\bar{x}_1 \vee \bar{x}_2 \vee x_3)(\bar{x}_1 \vee x_2 \vee \bar{x}_4)$$

$(x_1 \vee x_2 \vee \bar{x}_3)(\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)(x_1 \vee \bar{x}_2 \vee x_4)(x_1 \vee \bar{x}_2 \vee \bar{x}_4)$

ϕ_2 is not satisfiable

13

NP-Completeness (cont)

3SAT \leq_p CLIQUE

Step 1: Construct $f : \{\phi \mid 3\text{SAT formula}\} \rightarrow \{(G, k)\}$

Given $\phi = C_1 C_2 \cdots C_m$ where $C_r = (l_1^r \vee l_2^r \vee l_3^r)$

Let $G = (V, E)$ where $V = \bigcup_{r=1}^m \{v_1^r, v_2^r, v_3^r\}$

$E = \{(v_i^r, v_j^s) \mid r \neq s \text{ and } l_i^r \neq \bar{l}_j^s\}$

Let $k = m$

14

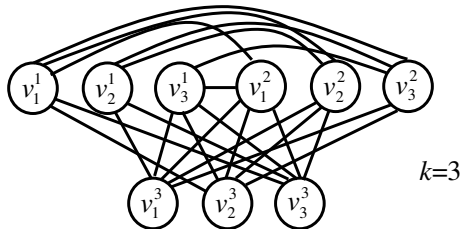
NP-Completeness (cont)

3SAT \leq_p CLIQUE

Example of construction

$$\phi = (x_1 \vee x_2 \vee \bar{x}_3)(\bar{x}_2 \vee x_3 \vee \bar{x}_4)(\bar{x}_1 \vee \bar{x}_2 \vee x_4)$$

$$(l_1^1 \vee l_2^1 \vee l_3^1)(l_1^2 \vee l_2^2 \vee l_3^2)(l_1^3 \vee l_2^3 \vee l_3^3)$$



15

NP-Completeness (cont)

3SAT \leq_p CLIQUE

Step 2: Show that $x \in L_{3\text{SAT}} \Leftrightarrow f(x) \in L_{\text{CLIQUE}}$

ϕ is satisfiable $\Leftrightarrow f(\phi) = (G, k)$ where G has a clique of size $\geq k$

Proof : Suppose ϕ is satisfiable.

Then $\exists \bar{x}$ such that $\phi(\bar{x}) = 1$.

Under \bar{x} each clause of ϕ has at least one literal whose value is 1.

From each clause $C_r = (l_1^r \vee l_2^r \vee l_3^r)$ of ϕ pick a literal l_i^r whose value is 1 under \bar{x} .

16

NP-Completeness (cont)

3SAT \leq_p CLIQUE Proof : (continued)

Let $V' = \{v_{i_1}^1, v_{i_2}^2, \dots, v_{i_m}^m\}$ be the m vertices corresponding to the m literals $l_{i_j}^j$ just picked.

No pair of vertices of V' correspond to literals x_j and \bar{x}_j since all of the literals corresponding to V' are 1 under \bar{x} .

Each pair of vertices of V' share an edge since they belong to different clauses and are not the negation of each other.

V' is a clique of size m .

$f(\phi) = (G, k)$ where G has a clique of size $\geq k = m$

17

NP-Completeness (cont)

3SAT \leq_p CLIQUE Proof : (continued)

Now suppose $f(\phi) = (G, k)$ and G has a clique of size $\geq k = m$

Let V' be a clique of size $\geq m$ in G .

Since vertices from the same clause do not share edges

V' must have exactly m vertices one from each clause.

The m literals corresponding to V' do not contain both the negated and unnegated form of a variable since their corresponding vertices in V' would not share an edge.

18

NP-Completeness (cont)

3SAT \leq_p CLIQUE Proof : (continued)

Construct a truth assignment \bar{x} for ϕ by setting variable $x_j = 1$ if x_j appears unnegated in the literals corresponding to V' and $x_j = 0$ otherwise.

By construction all literals corresponding to V' are 1 under the truth assignment \bar{x} .

Since each clause of ϕ contains a literal corresponding to a vertex in V' , $\phi(\bar{x}) = 1$.

ϕ is satisfiable.

QED

19

NP-Completeness (cont)

3SAT \leq_p CLIQUE

Step 3: Show that f is computable in polynomial time.

If ϕ has m clauses and n variables, then $f(\phi) = (G, m)$ where G has $3m$ vertices and at most $9m(m-1)/2$ edges.

Whether two vertices of G share an edge or not can be determined by inspection of ϕ .

f is computable in polynomial time.

3SAT \leq_p CLIQUE has been shown.

20