

Complexity Theory

Lecture 6

Anuj Dawar

University of Cambridge Computer Laboratory
Easter Term 2015

<http://www.cl.cam.ac.uk/teaching/1415/Complexity/>

Composing Reductions

Polynomial time reductions are clearly closed under composition.

So, if $L_1 \leq_P L_2$ and $L_2 \leq_P L_3$, then we also have $L_1 \leq_P L_3$.

Note, this is also true of \leq_L , though less obvious.

If we show, for some problem A in NP that

$$\text{SAT} \leq_P A$$

or

$$3\text{SAT} \leq_P A$$

it follows that A is also NP -complete.

Independent Set

Given a graph $G = (V, E)$, a subset $X \subseteq V$ of the vertices is said to be an *independent set*, if there are no edges (u, v) for $u, v \in X$.

The natural algorithmic problem is, given a graph, find the largest independent set.

To turn this *optimisation problem* into a *decision problem*, we define **IND** as:

The set of pairs (G, K) , where G is a graph, and K is an integer, such that G contains an independent set with K or more vertices.

IND is clearly in **NP**. We now show it is **NP**-complete.

Reduction

We can construct a reduction from 3SAT to IND.

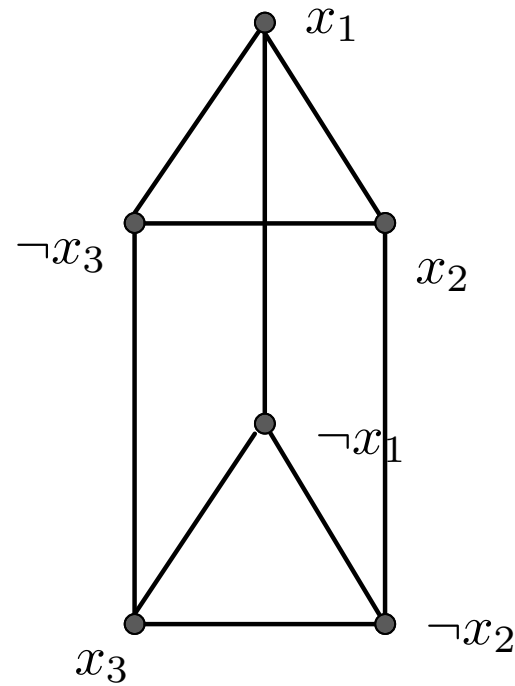
A Boolean expression ϕ in 3CNF with m clauses is mapped by the reduction to the pair (G, m) , where G is the graph obtained from ϕ as follows:

G contains m triangles, one for each clause of ϕ , with each node representing one of the literals in the clause.

Additionally, there is an edge between two nodes in different triangles if they represent literals where one is the negation of the other.

Example

$$(x_1 \vee x_2 \vee \neg x_3) \wedge (x_3 \vee \neg x_2 \vee \neg x_1)$$



Clique

Given a graph $G = (V, E)$, a subset $X \subseteq V$ of the vertices is called a *clique*, if for every $u, v \in X$, (u, v) is an edge.

As with **IND**, we can define a decision problem:

CLIQUE is defined as:

The set of pairs (G, K) , where G is a graph, and K is an integer, such that G contains a clique with K or more vertices.

Clique 2

CLIQUE is in NP by the algorithm which *guesses* a clique and then verifies it.

CLIQUE is NP-complete, since

$IND \leq_P \text{CLIQUE}$

by the reduction that maps the pair (G, K) to (\bar{G}, K) , where \bar{G} is the complement graph of G .

k -Colourability

A graph $G = (V, E)$ is k -colourable, if there is a function

$$\chi : V \rightarrow \{1, \dots, k\}$$

such that, for each $u, v \in V$, if $(u, v) \in E$,

$$\chi(u) \neq \chi(v)$$

This gives rise to a decision problem for each k .

2-colourability is in \mathbf{P} .

For all $k > 2$, k -colourability is \mathbf{NP} -complete.

3-Colourability

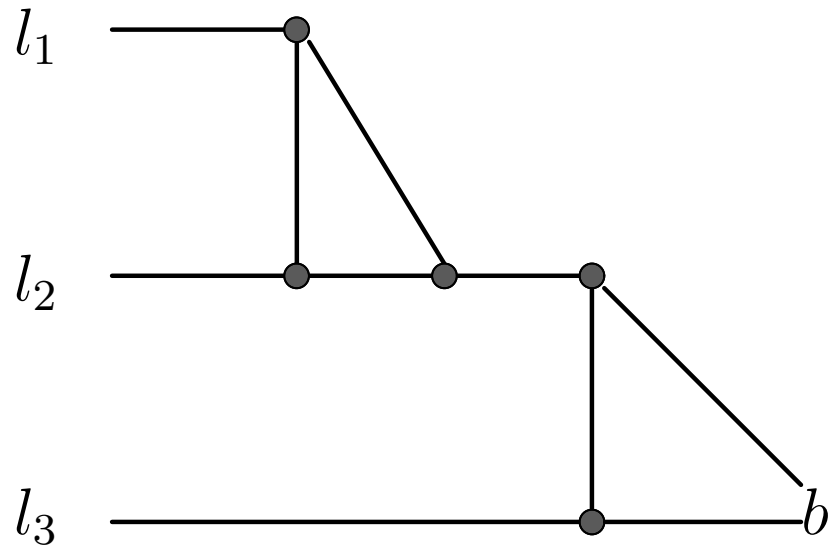
3-Colourability is in NP, as we can *guess* a colouring and verify it.

To show NP-completeness, we can construct a reduction from 3SAT to 3-Colourability.

For each variable x , we have two vertices x, \bar{x} which are connected in a triangle with the vertex a (common to all variables).

In addition, for each clause containing the literals l_1, l_2 and l_3 we have a gadget.

Gadget



With a further edge from a to b .

Hamiltonian Graphs

Recall the definition of **HAM**—the language of Hamiltonian graphs.

Given a graph $G = (V, E)$, a *Hamiltonian cycle* in G is a path in the graph, starting and ending at the same node, such that every node in V appears on the cycle *exactly once*.

A graph is called *Hamiltonian* if it contains a Hamiltonian cycle.

The language **HAM** is the set of encodings of Hamiltonian graphs.

Hamiltonian Cycle

We can construct a reduction from **3SAT** to **HAM**

Essentially, this involves coding up a Boolean expression as a graph, so that every satisfying truth assignment to the expression corresponds to a Hamiltonian circuit of the graph.

This reduction is much more intricate than the one for **IND**.

Travelling Salesman

Recall the travelling salesman problem

Given

- V — a set of nodes.
- $c : V \times V \rightarrow \mathbb{N}$ — a cost matrix.

Find an ordering v_1, \dots, v_n of V for which the total cost:

$$c(v_n, v_1) + \sum_{i=1}^{n-1} c(v_i, v_{i+1})$$

is the smallest possible.

Travelling Salesman

As with other optimisation problems, we can make a decision problem version of the Travelling Salesman problem.

The problem **TSP** consists of the set of triples

$$(V, c : V \times V \rightarrow \mathbb{N}, t)$$

such that there is a tour of the set of vertices V , which under the cost matrix c , has cost t or less.

Reduction

There is a simple reduction from **HAM** to **TSP**, mapping a graph (V, E) to the triple $(V, c : V \times V \rightarrow \mathbb{N}, n)$, where

$$c(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 2 & \text{otherwise} \end{cases}$$

and n is the size of V .

Sets, Numbers and Scheduling

It is not just problems about formulas and graphs that turn out to be **NP**-complete.

Literally hundreds of naturally arising problems have been proved **NP**-complete, in areas involving network design, scheduling, optimisation, data storage and retrieval, artificial intelligence and many others.

Such problems arise naturally whenever we have to construct a solution within constraints, and the most effective way appears to be an exhaustive search of an exponential solution space.

We now examine three more **NP**-complete problems, whose significance lies in that they have been used to prove a large number of other problems **NP**-complete, through reductions.

3D Matching

The decision problem of *3D Matching* is defined as:

Given three disjoint sets X , Y and Z , and a set of triples $M \subseteq X \times Y \times Z$, does M contain a matching?

I.e. is there a subset $M' \subseteq M$, such that each element of X , Y and Z appears in exactly one triple of M' ?

We can show that **3DM** is **NP**-complete by a reduction from **3SAT**.