Complexity Theory

Lecture 5: NP-Complete Problems

Tom Gur

Preface: What do professors do all day?

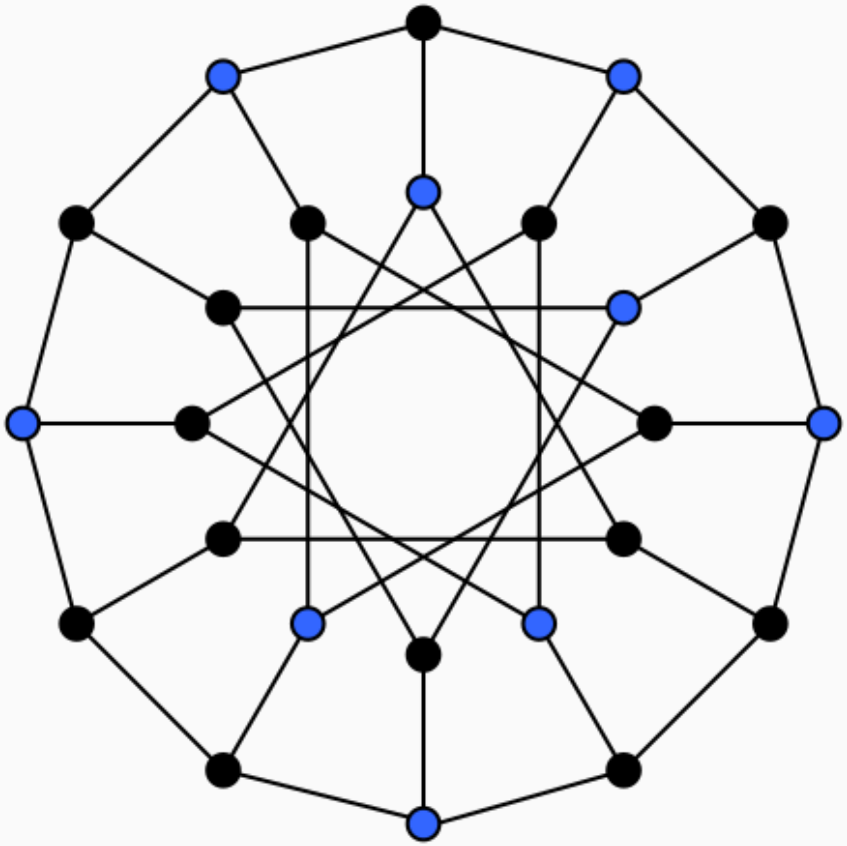
Recap

- $\mathcal{P}$ captures polynomial-time computation.
- $\mathcal{NP}$ captures polynomial-time verification.
- A problem is $\mathcal{NP}$-hard if any language in $\mathcal{NP}$ is reducible to it.
- A problem is $\mathcal{NP}$-complete if it is: (1) $\mathcal{NP}$-hard, (2) in $\mathcal{NP}$.
- Cook-Levin Theorem: SAT is $\mathcal{NP}$-complete.
- In fact, so is CNF-SAT.
- And CNF-SAT is reducible to 3SAT:
  $(x_1 \vee x_2 \vee x_3 \vee x_4) \rightarrow (x_1 \vee x_2 \vee z_1) \wedge (\neg z_1 \vee x_3 \vee z_2) \wedge (\neg z_2 \vee x_4)$

Let's see some reductions!

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 $u, v \in X$.



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 IS 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.*

IS is clearly in NP. We now show it is NP-complete.

Reduction

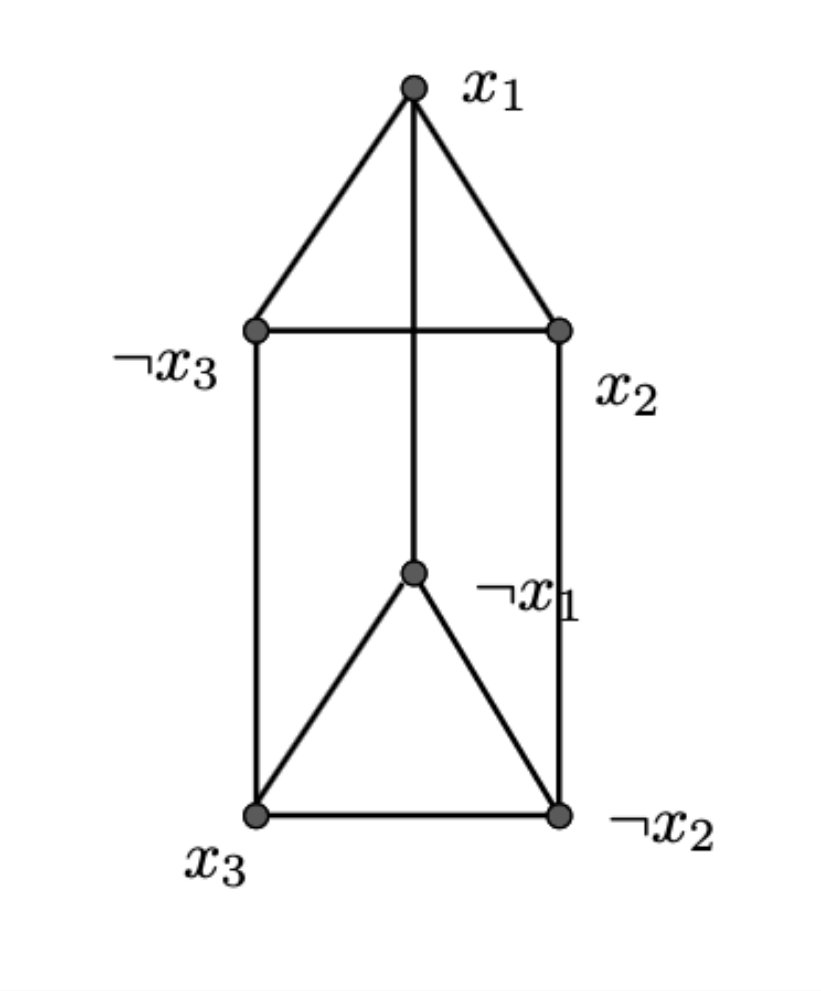
We can construct a reduction from 3SAT to IS.

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

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



Proof

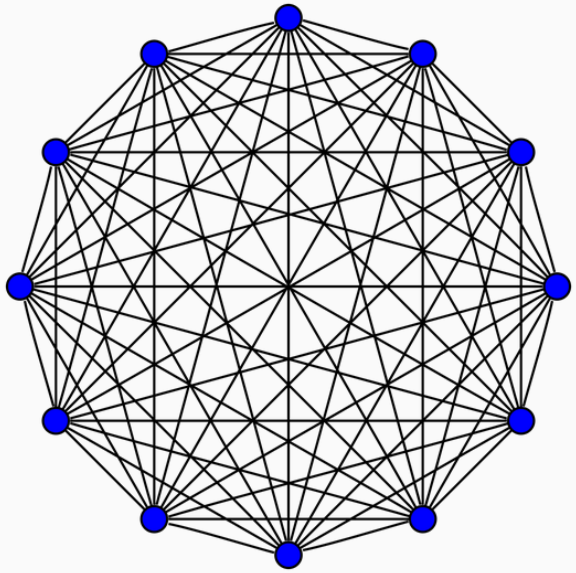
Efficiency: The transformation is computable in polynomial time: write down all triangles and connect literals.

Completeness: Suppose ϕ is satisfiable, and let T be a satisfying assignment. Choose a satisfied literal from each of the m clauses. Denote the corresponding set of vertices by X, and observe that: (1) X does not contain a variable and its negation, and (2) X does not contains two veritices from the same triange. Hence X is IS of size m

Soundness: Suppose ϕ is unsatisfiable, and towards contradiction, suppose X is IS of size m. Note that X must contain at most 1 vertex per triangle and no edges between triangles. Hence its corresponding literals form a satisfying assignment of ϕ, in contradiction.

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 IS, 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

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

CLIQUE is NP-complete, since

IS $\leq_P$ CLIQUE

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

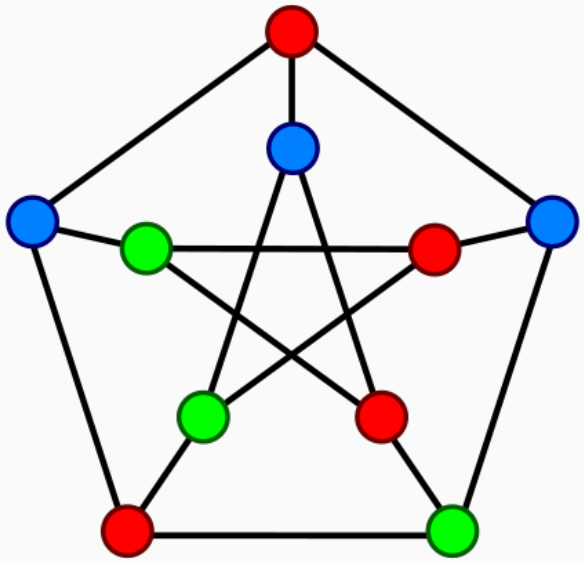
k-Colourability

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

$$\chi : V \to \{1, \ldots, k\}$$

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

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



k-Colourability

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

$$\chi : V \to \{1, \ldots, k\}$$

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 P. (How to intimidate your Google interviewer...)

For all $k > 2$, k-colourability is NP-complete.

3-Colourability

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

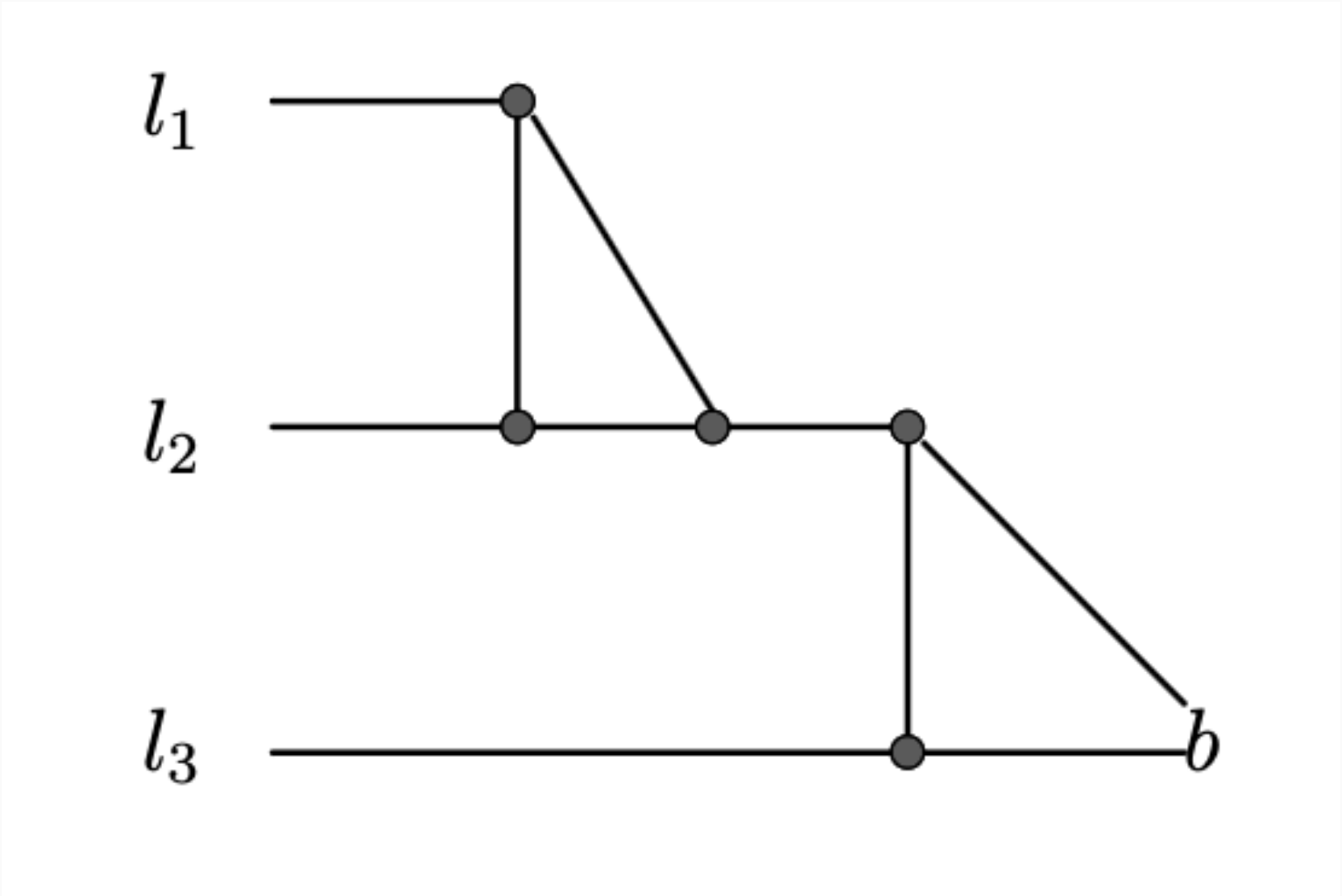
To show NP-completeness, we can construct a reduction from 3SAT to 3-Colourability.

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.

With a further edge from a to b.

Gadget

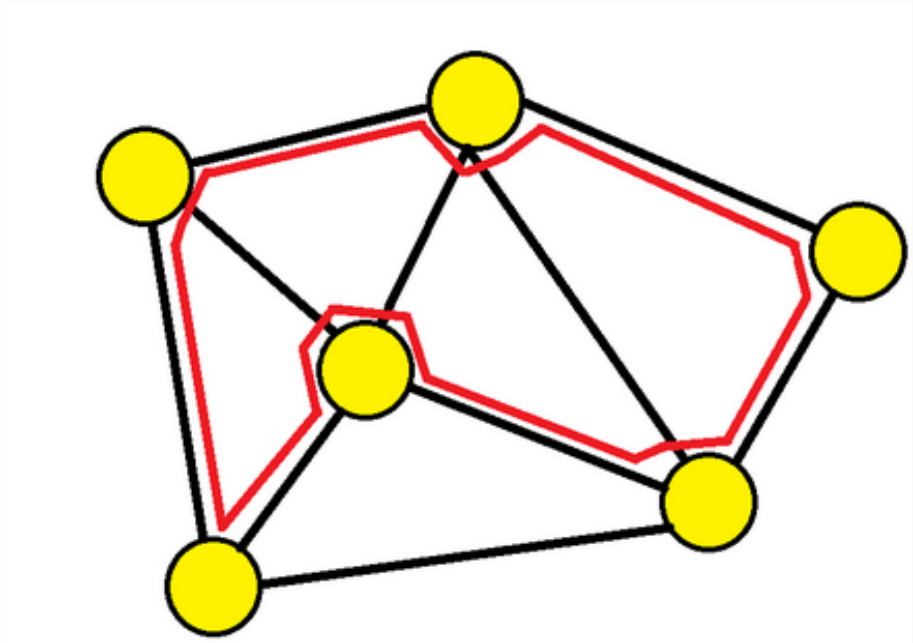


Turing fact of the day

Alan Turing used to cycle to work in a gas mask during the first week of June each year to combat hay fever...

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*.



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.

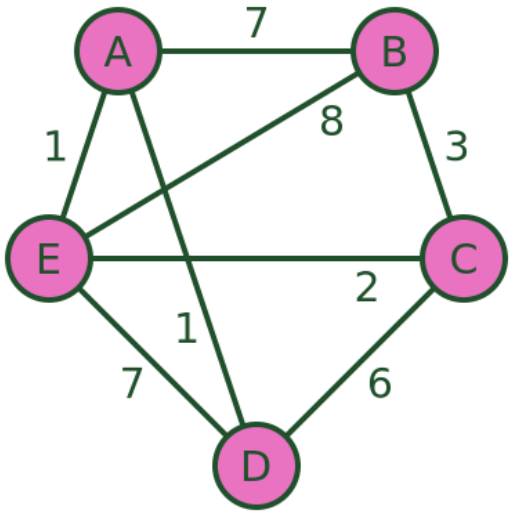
This reduction is much more intricate than the one for IS.

Travelling Salesman

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

Find an ordering $v_1, \ldots, v_n$ of V minimising:

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



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

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

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 $(V, c : V \times V \to \mathbb{N}, n)$, where

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

and n is the size of V.