NP and NP-Completeness

Design and Analysis of Algorithms

Andrei Bulatov

Efficient Certification

By a "solution" of a decision problem X we understand a certificate witnessing that an instance is a "yes"-instance

We say that an algorithm B is an efficient certifier for a problem X if

- B is a polynomial time algorithm that takes two input arguments: instance s and a certificate t
- there is polynomial p such that for every string s, we have $s \in X$ if and only if there exists a string t such that $|t| \leq p(|s|)$ and $B(s,t) = yes$

The class of problems having an efficient certifier is denoted by NP

Certifying vs. Solving

Certifying and brute force

Efficient Certification: Composite

COMPOSITES. Given an integer s, is s composite?

Certificate. A nontrivial factor t of s. Note that such a certificate exists iff s is composite. Moreover $|t| \leq |s|$.

Certifier.

> Check if $t > 1$ and $t < s$
> If yes, check if t is a divisor of s

Instance. s = 437,669.

Certificate. t = 541 or 809. $437{,}669 = 541 \times 809$

Conclusion. COMPOSITES is in NP.

Efficient Certification: 3-SAT

SAT. Given a CNF formula Φ, is there a satisfying assignment?

Certificate. An assignment of truth values to the n boolean variables.

Certifier. Check that each clause in Φ has at least one true literal.

Example

$$(\overline{x_1} \vee x_2 \vee x_3) \wedge (x_1 \vee \overline{x_2} \vee x_3) \wedge (x_1 \vee x_2 \vee x_4) \wedge (\overline{x_1} \vee \overline{x_3} \vee \overline{x_4})$$

instance s

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

certificate t

Conclusion. SAT is in NP.

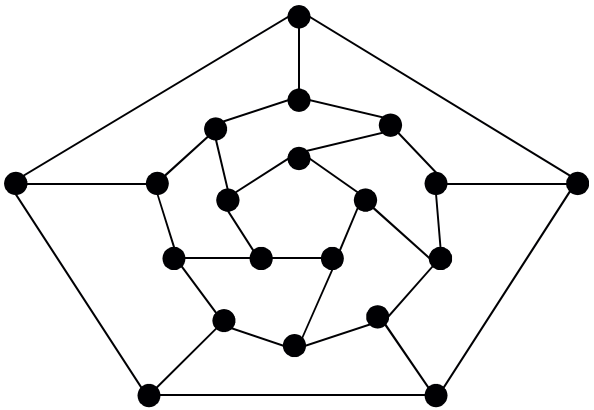
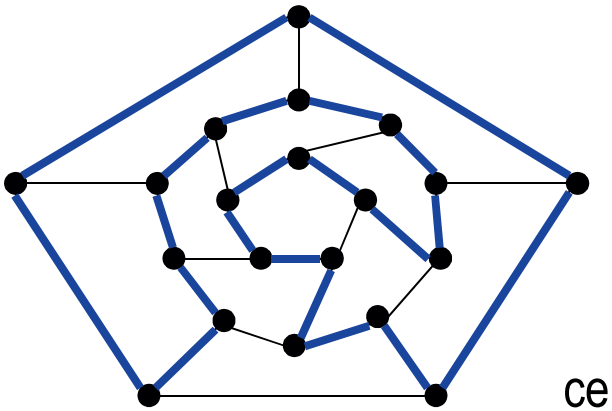
Efficient Certification: Hamilton Cycle

HAM-CYCLE. Given an undirected graph G = (V, E), does there exist a simple cycle C that visits every node?

Certificate. A permutation of the n nodes.

Certifier. Check that the permutation contains each node in V exactly once, and that there is an edge between each pair of adjacent nodes in the permutation.

Conclusion. HAM-CYCLE is in NP.

instance s   certificate t

NP

P is the class of problems for which there is a polynomial time algorithm

NP is the class of problems for which there is an efficient certifier

3-SAT, Independent Set, Vertex Cover, problems about feasible circulations are in NP

Lemma

$P \subseteq NP$

Proof

Certifier is a solution algorithm that runs with empty certificate.

NP-Completeness

What are the most difficult problems in NP?

A problem X is said to be NP-complete if

(i) $X \in$ NP

(ii) for any $Y \in$ NP, we have $Y \leq X$

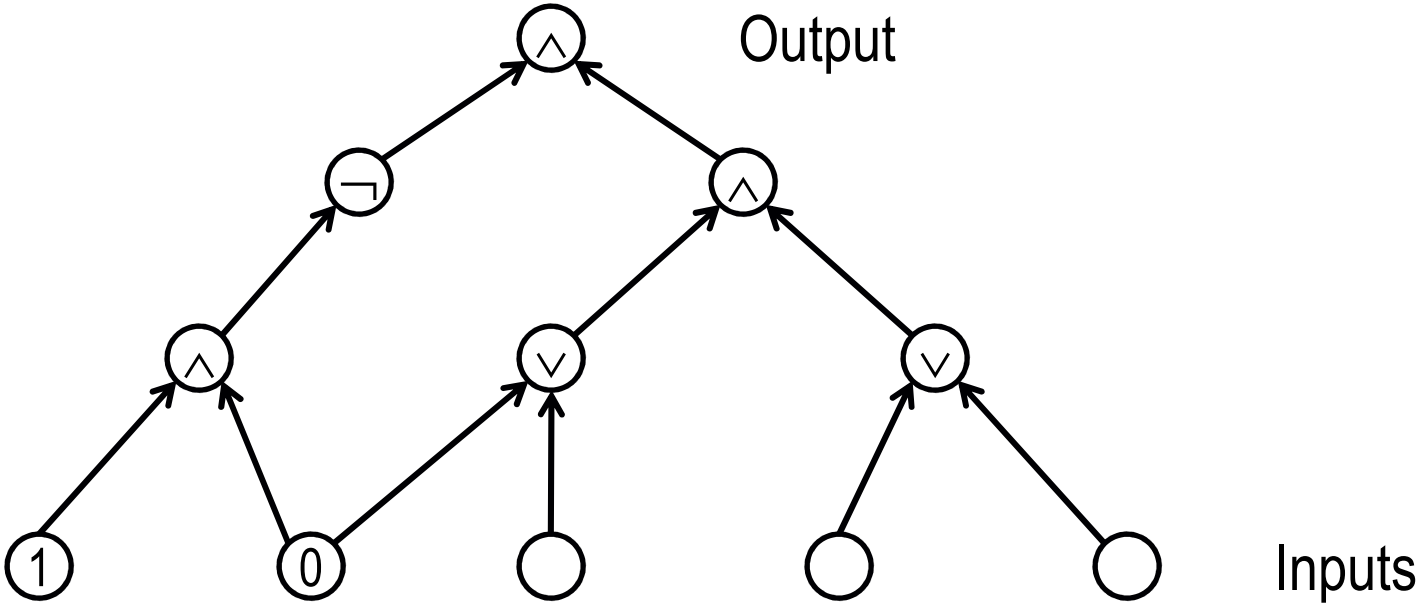
Lemma

If an NP-complete problem solvable in polynomial time then P = NP.

Circuit Satisfiability

A circuit consists of:

- inputs
- wires
- logic gates: $\wedge$ (AND), $\vee$ (OR), $\neg$ (NOT)
- output



Circuit Satisfiability (cntd)

The output computed by a circuit is defined in the natural way

A circuit is said to be satisfiable if there are values of the inputs such that the output is 1

The Circuit Satisfiability Problem

Instance:

A circuit C

Objective:

Is C satisfiable?

Circuit Satisfiability: NP-Completeness

Theorem (Cook, Levin)
Circuit Satisfiability is NP-complete

Proof (Idea)

We have to reduce every problem $X \in$ NP to Circuit Satisfiability

Use the fact that X has an efficient certifier $B(\cdot,\cdot)$

The main idea is that the work of any algorithm on inputs of fixed length can be simulated by a circuit

Simulation is in the sense that there is a circuit that outputs 1 if and only if the algorithm outputs "yes"

Moreover, the number of gates (size) of the circuit is O(running time of the algorithm)

Circuit Satisfiability: NP-Completeness (cntd)

In order to decide if $s \in X$, we have to check if there is a string t of length $p(|s|)$ such that $B(s,t)$ outputs "yes"

We use Circuit Satisfiability as a black box as follows:

Consider $B(\cdot,\cdot)$ as an algorithm on $n + p(n)$ bits

Transform $B(s,\cdot)$ into a circuit $C(s)$ with s `hardwired', and $p(|s|)$ inputs for possible t

Ask if $C(s)$ is satisfiable.

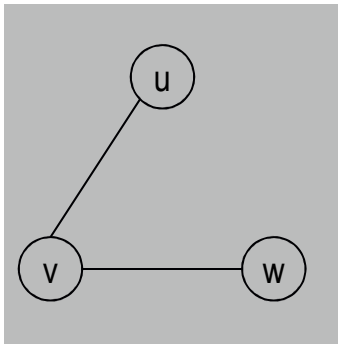
If yes, there is a required t, and therefore $s \in X$

If not, there is no t such that $B(s,t)$ = "yes", hence $s \notin X$

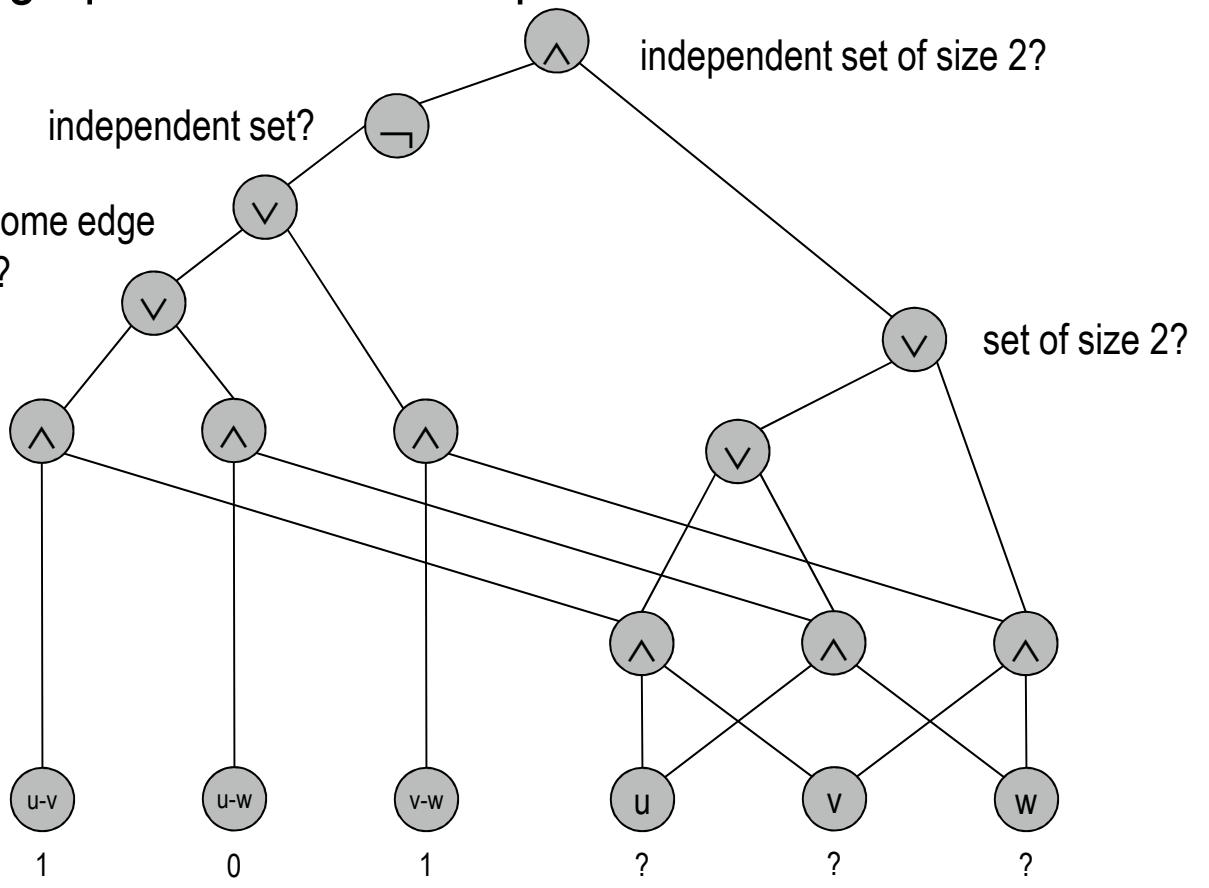
QED

Example

Construction below creates a circuit C whose inputs can be set so that C outputs 1 iff graph G has an independent set of size 2.



G = (V, E), n = 3



$\binom{n}{2}$ hard-coded inputs (graph description)

n inputs (nodes in independent set)

Proving NP-Completeness

Remark. Once we establish first "natural" NP-complete problem, others are much easier

Recipe to establish NP-completeness of problem Y.

Step 1. Show that Y is in NP.

Step 2. Choose an NP-complete problem X.

Step 3. Prove that $X \leq Y$.

Proving NP-Completeness

Lemma

If X is an NP-complete problem, and Y is a problem in NP with the property that $X \leq Y$ then Y is NP-complete.

Proof

Let W be any problem in NP. Then $W \leq X \leq Y$.

By transitivity, $W \leq Y$.

Hence Y is NP-complete.

QED

3-SAT: NP-Completeness

Theorem

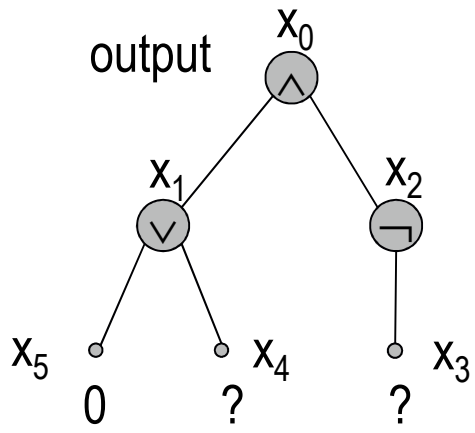
3-SAT is NP-complete

Proof

Suffices to show that CIRCUIT-SAT $\leq$ 3-SAT since 3-SAT is in NP.

Let C be any circuit.

Create a 3-SAT variable x_i for each circuit element i.



3-SAT: NP-Completeness

Make circuit compute correct values at each node:

$x_2 = \neg x_3 \Rightarrow$ add 2 clauses: $x_2 \vee x_3$, $\overline{x_2} \vee \overline{x_3}$

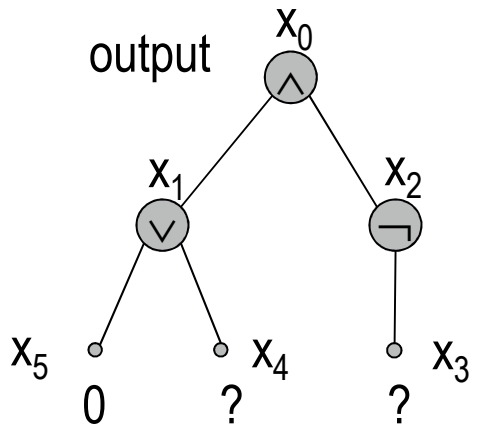
$x_1 = x_4 \vee x_5 \Rightarrow$ add 3 clauses: $x_1 \vee \overline{x_4}$, $x_1 \vee \overline{x_5}$, $\overline{x_1} \vee x_4 \vee x_5$

$x_0 = x_1 \wedge x_2 \Rightarrow$ add 3 clauses: $\overline{x_0} \vee x_1$, $\overline{x_0} \vee x_2$, $x_0 \vee \overline{x_1} \vee \overline{x_2}$

Hard-coded input values and output value.

$x_5 = 0 \Rightarrow$ add 1 clause: $\overline{x_5}$

$x_0 = 1 \Rightarrow$ add 1 clause: x_0



Final step: turn clauses of length < 3 into clauses of length exactly 3.

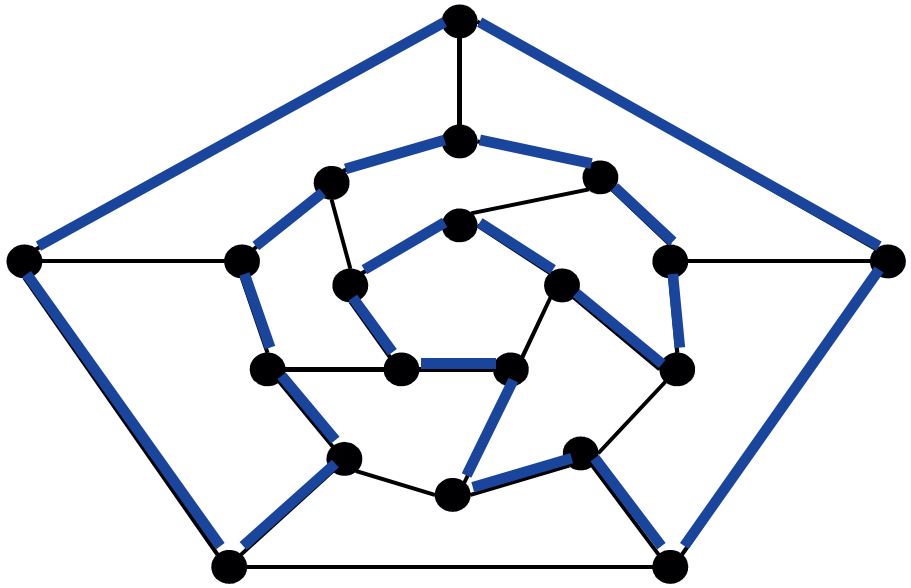
QED

Hamiltonian Cycle

The Hamiltonian Cycle Problem

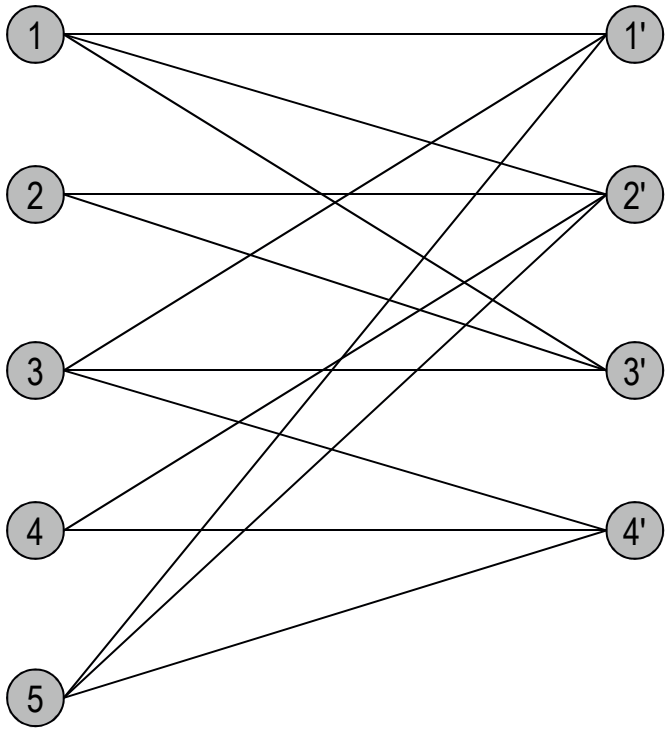
Instance: An undirected graph G = (V, E)

Objective: Does there exist a simple cycle Γ that contains every node in V.



YES: vertices and faces of a dodecahedron.

Hamiltonian Cycle



NO: bipartite graph with odd number of nodes.

Directed Hamiltonian Cycle

The Directed Hamiltonian Cycle Problem

Instance: A directed graph G = (V, E)

Objective: Does there exist a simple directed cycle Γ that contains every node in V.

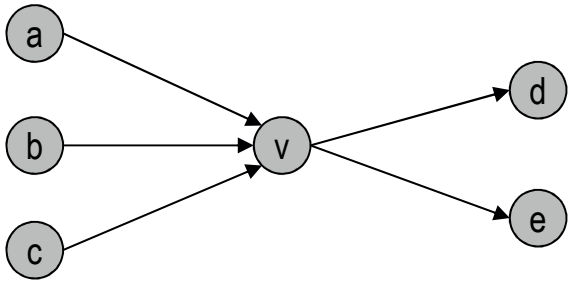
Lemma

Directed Hamiltonian Cycle $\leq$ Hamiltonian Cycle.

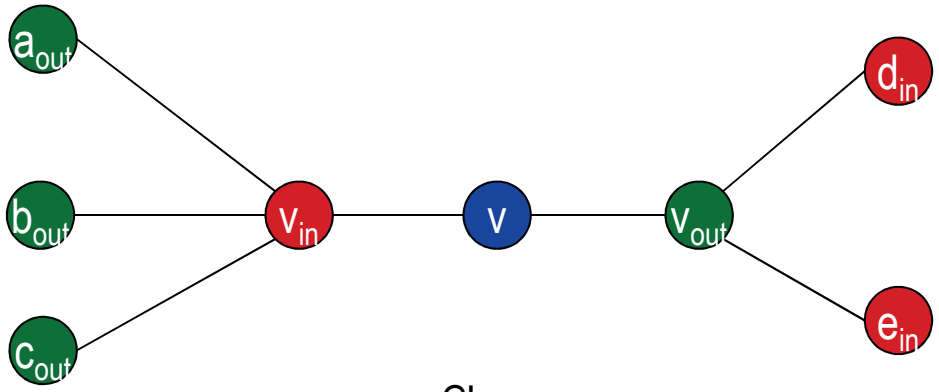
Proof

Given a directed graph G = (V, E) with n nodes, construct an undirected graph G' with 3n nodes.

Directed Hamiltonian Cycle



G



G'

We show that G has a Hamiltonian cycle iff G' does.

$\Rightarrow$

Suppose G has a directed Hamiltonian cycle Γ.

Then G' has an undirected Hamiltonian cycle (same order).

Directed Hamiltonian Cycle

$\Leftarrow$

Suppose G' has an undirected Hamiltonian cycle Γ'.

Γ' must visit nodes in G' using one of following two orders:

…, B, G, R, B, G, R, B, G, R, B, …

…, B, R, G, B, R, G, B, R, G, B, …

Blue nodes in Γ' make up directed Hamiltonian cycle Γ in G, or reverse of one.

QED

Ham-Cycle: NP-Completeness

Theorem

3-SAT $\leq$ Directed Hamiltonian Cycle

Proof

Given an instance Φ of 3-SAT, we construct an instance of Directed Hamiltonian Cycle that has a Hamiltonian cycle iff Φ is satisfiable.

Construction.

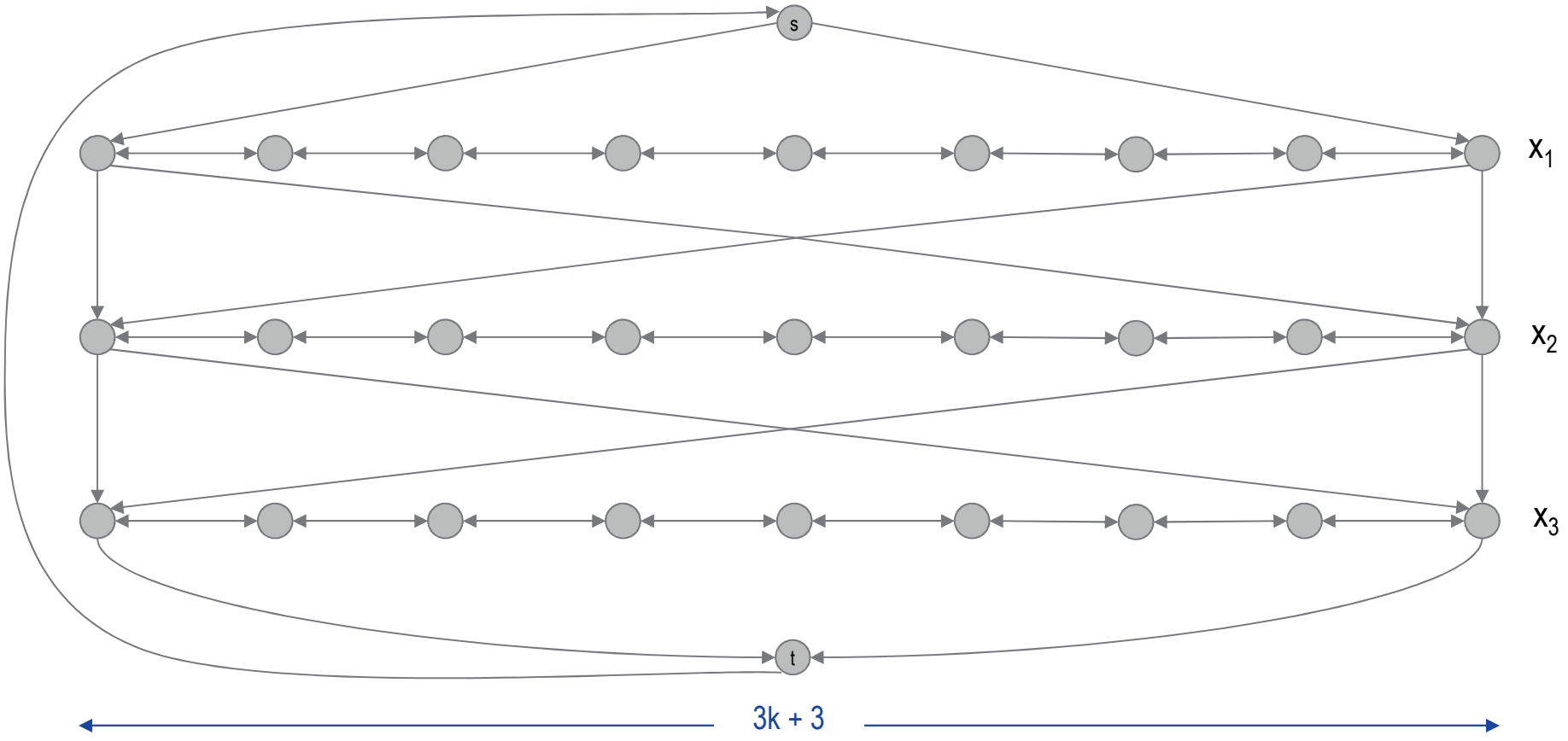
First, create graph that has 2^n Hamiltonian cycles which correspond in a natural way to 2^n possible truth assignments.

Given 3-SAT instance Φ with n variables x_i and k clauses.

Construct G to have 2n Hamiltonian cycles.

Intuition: traverse path i from left to right $\Leftrightarrow$ set variable $x_i = 1$

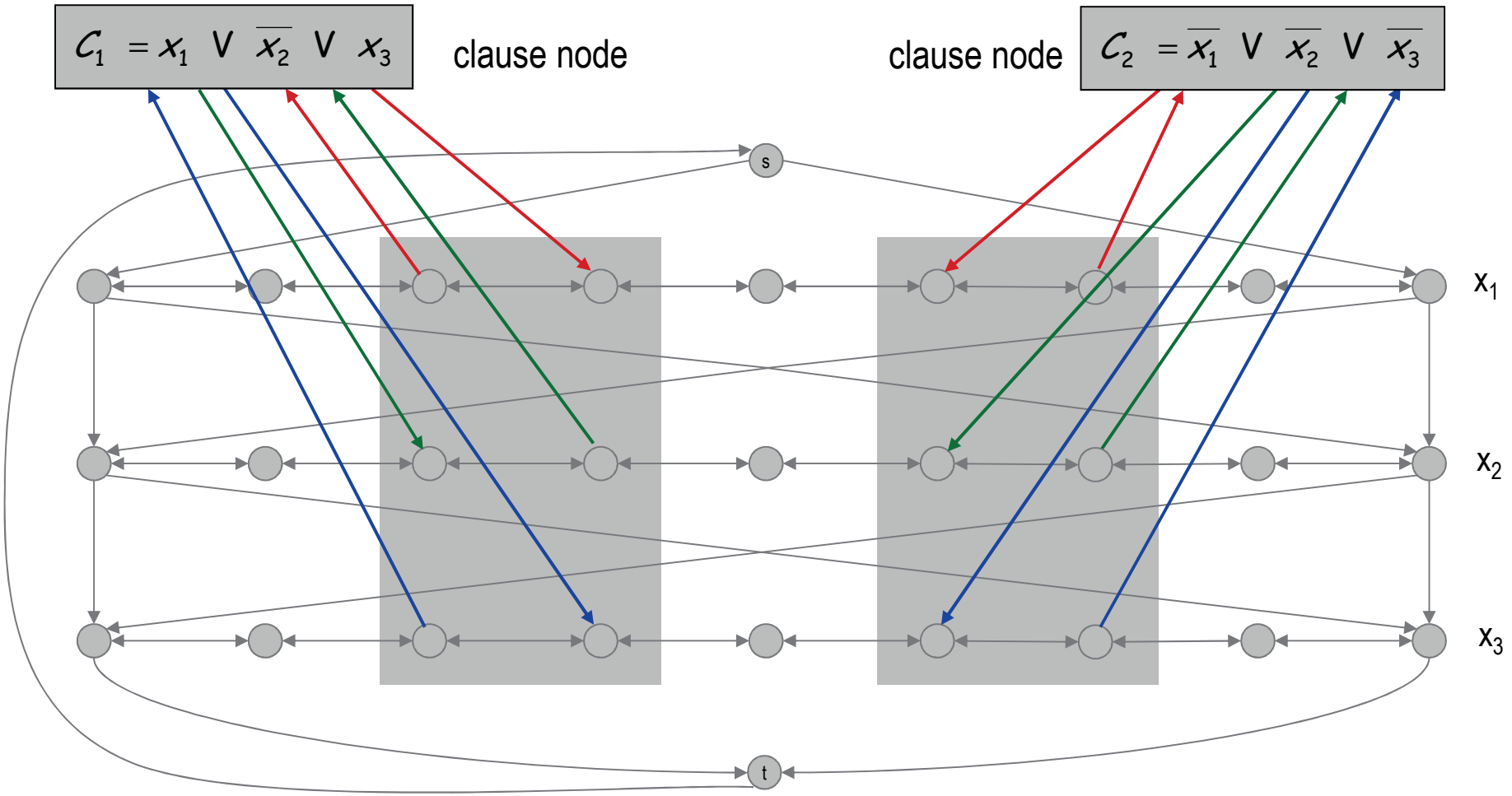
Ham-Cycle: NP-Completeness



Ham-Cycle: NP-Completeness

Given 3-SAT instance Φ with n variables x_i and k clauses.

For each clause: add a node and 6 edges.



Ham-Cycle: NP-Completeness

Claim.

Φ is satisfiable iff G has a Hamiltonian cycle.

$\Rightarrow$

Suppose 3-SAT instance has satisfying assignment x*.

Then, define Hamiltonian cycle in G as follows:

if $x_i^* = 1$, traverse row i from left to right

if $x_i^* = 0$, traverse row i from right to left

for each clause C_i, there will be at least one row i in which we are going in "correct" direction to splice node C_i into tour

Proving NP-Completeness

$\Leftarrow$

Suppose G has a Hamiltonian cycle Γ.

If Γ enters clause node C_i, it must depart on mate edge.

Thus, nodes immediately before and after C_i are connected by an edge e in G

removing C_i from cycle, and replacing it with edge e yields Hamiltonian cycle on G – { C_i }

Continuing in this way, we are left with Hamiltonian cycle Γ' in G – { $C_1, C_2, \ldots, C_k$ }.

Set $x_i^* = 1$ iff Γ' traverses row i left to right.

Since Γ visits each clause node C_i, at least one of the paths is traversed in "correct" direction, and each clause is satisfied.