

Theory of Computer Science

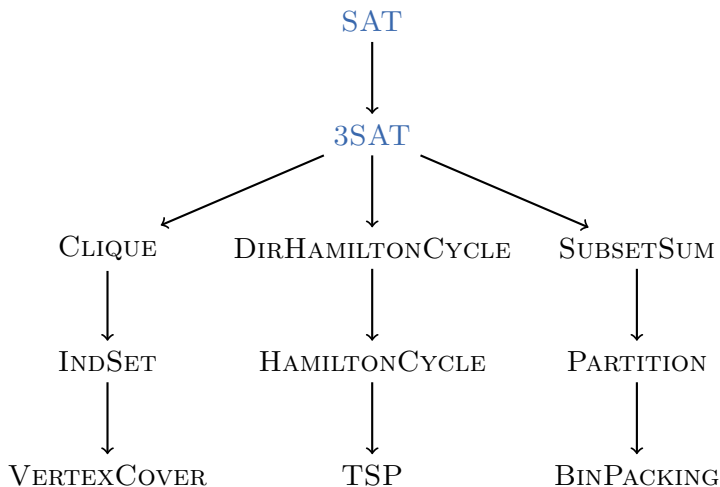
D4. Some NP-Complete Problems, Part I

Gabriele Röger

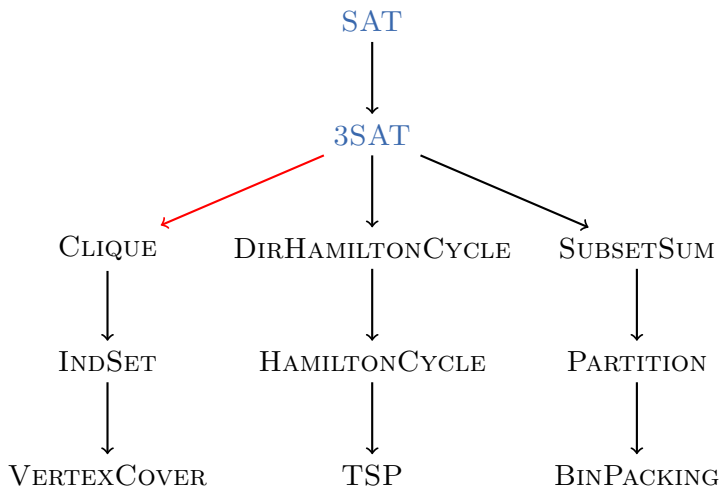
University of Basel

May 18, 2020

Overview of the Reductions



Graph Problems

$$3\text{SAT} \leq_p \text{CLIQUE}$$


CLIQUE

Definition (CLIQUE)

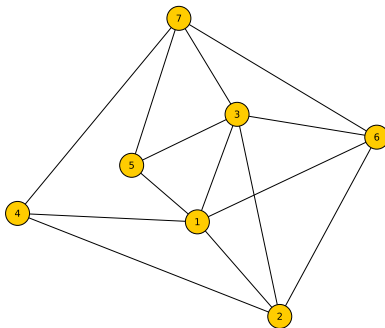
The problem **CLIQUE** is defined as follows:

Given: undirected graph $G = \langle V, E \rangle$, number $K \in \mathbb{N}_0$

Question: Does G have a clique of size at least K ,
i. e., a set of vertices $C \subseteq V$ with $|C| \geq K$
and $\{u, v\} \in E$ for all $u, v \in C$ with $u \neq v$?

German: Clique

Cliques: Exercise (slido)



How many nodes has the largest clique of this graph?



CLIQUE is NP-Complete (1)

Theorem (CLIQUE is NP-Complete)

CLIQUE *is NP-complete*.

CLIQUE is NP-Complete (2)

Proof.

CLIQUE $\in$ NP: guess and check.

CLIQUE is NP-Complete (2)

Proof.

CLIQUE $\in$ NP: guess and check.

CLIQUE is NP-hard: We show $3\text{SAT} \leq_p \text{CLIQUE}$.

CLIQUE is NP-Complete (2)

Proof.

CLIQUE $\in$ NP: guess and check.

CLIQUE is NP-hard: We show $3\text{SAT} \leq_p \text{CLIQUE}$.

- We are given a 3-CNF formula φ , and we may assume that each clause has exactly three literals.
- In polynomial time, we must construct a graph $G = \langle V, E \rangle$ and a number K such that:
 G has a clique of size at least K iff φ is satisfiable.

CLIQUE is NP-Complete (2)

Proof.

CLIQUE $\in$ NP: guess and check.

CLIQUE is NP-hard: We show $3\text{SAT} \leq_p \text{CLIQUE}$.

- We are given a 3-CNF formula φ , and we may assume that each clause has exactly three literals.
- In polynomial time, we must construct a graph $G = \langle V, E \rangle$ and a number K such that:
 G has a clique of size at least K iff φ is satisfiable.

↪ construction of V, E, K on the following slides.

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

- $V = \{\langle i, j \rangle \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$
 $\rightsquigarrow$ a vertex for every literal of every clause

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

- $V = \{\langle i, j \rangle \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$
 $\rightsquigarrow$ a vertex for every literal of every clause
- E contains edge between $\langle i, j \rangle$ and $\langle i', j' \rangle$ if and only if
 - $i \neq i' \rightsquigarrow$ belong to **different clauses**, and
 - ℓ_{ij} and $\ell_{i'j'}$ are **not complementary literals**

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

- $V = \{\langle i, j \rangle \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$
 $\rightsquigarrow$ a vertex for every literal of every clause
- E contains edge between $\langle i, j \rangle$ and $\langle i', j' \rangle$ if and only if
 - $i \neq i' \rightsquigarrow$ belong to **different clauses**, and
 - ℓ_{ij} and $\ell_{i'j'}$ are **not complementary literals**
- $K = m$

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

- $V = \{\langle i, j \rangle \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$
 $\rightsquigarrow$ a vertex for every literal of every clause
- E contains edge between $\langle i, j \rangle$ and $\langle i', j' \rangle$ if and only if
 - $i \neq i' \rightsquigarrow$ belong to **different clauses**, and
 - ℓ_{ij} and $\ell_{i'j'}$ are **not complementary literals**
- $K = m$

$\rightsquigarrow$ obviously polynomially computable

CLIQUE is NP-Complete (3)

Proof (continued).

Let m be the number of clauses in φ .

Let ℓ_{ij} the j -th literal in clause i .

Define V , E , K as follows:

- $V = \{\langle i, j \rangle \mid 1 \leq i \leq m, 1 \leq j \leq 3\}$
 $\rightsquigarrow$ a vertex for every literal of every clause
- E contains edge between $\langle i, j \rangle$ and $\langle i', j' \rangle$ if and only if
 - $i \neq i' \rightsquigarrow$ belong to **different clauses**, and
 - ℓ_{ij} and $\ell_{i'j'}$ are **not complementary literals**
- $K = m$

$\rightsquigarrow$ obviously polynomially computable

to show: reduction property

...

CLIQUE is NP-Complete (4)

Proof (continued).

($\Rightarrow$): If φ is satisfiable, then $\langle V, E \rangle$ has clique of size at least K :

CLIQUE is NP-Complete (4)

Proof (continued).

($\Rightarrow$): If φ is satisfiable, then $\langle V, E \rangle$ has clique of size at least K :

- Given a satisfying variable assignment choose a vertex corresponding to a satisfied literal in each clause.
- The chosen K vertices are all connected with each other and hence form a clique of size K .

...

CLIQUE is NP-Complete (5)

Proof (continued).

$(\Leftarrow)$: If $\langle V, E \rangle$ has a clique of size at least K , then φ is satisfiable:

CLIQUE is NP-Complete (5)

Proof (continued).

($\Leftarrow$): If $\langle V, E \rangle$ has a clique of size at least K , then φ is satisfiable:

- Consider a given clique C of size at least K .
- The vertices in C must all correspond to different clauses (vertices in the same clause are not connected by edges).

$\rightsquigarrow$ exactly one vertex per clause is included in C

- Two vertices in C never correspond to complementary literals X and $\neg X$ (due to the way we defined the edges).

CLIQUE is NP-Complete (5)

Proof (continued).

($\Leftarrow$): If $\langle V, E \rangle$ has a clique of size at least K , then φ is satisfiable:

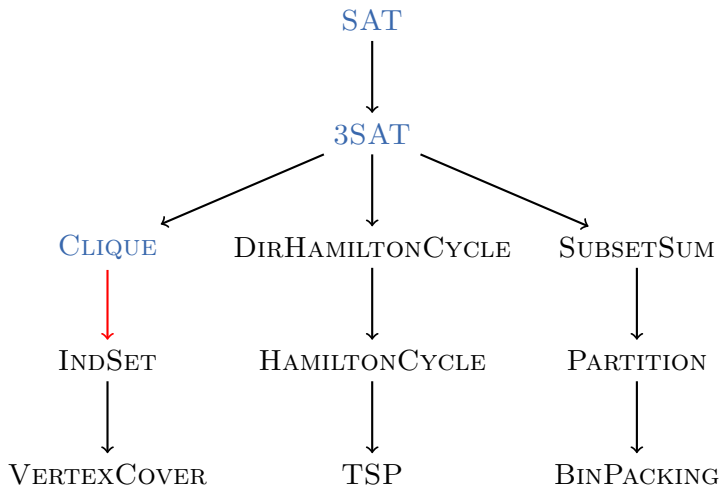
- Consider a given clique C of size at least K .
- The vertices in C must all correspond to different clauses (vertices in the same clause are not connected by edges).

$\rightsquigarrow$ exactly one vertex per clause is included in C

- Two vertices in C never correspond to complementary literals X and $\neg X$ (due to the way we defined the edges).
- If a vertex corresp. to X was chosen, map X to T (true).
- If a vertex corresp. to $\neg X$ was chosen, map X to F (false).
- If neither was chosen, arbitrarily map X to T or F.

$\rightsquigarrow$ satisfying assignment



$\text{CLIQUE} \leq_p \text{INDSET}$ 

INDSET

Definition (INDSET)

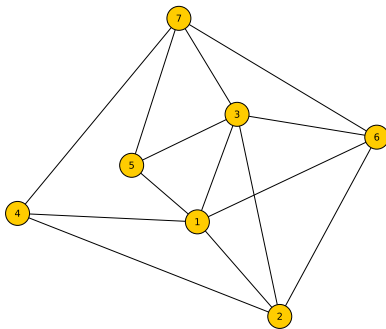
The problem **INDSET** is defined as follows:

Given: undirected graph $G = \langle V, E \rangle$, number $K \in \mathbb{N}_0$

Question: Does G have an independent set of size at least K ,
i. e., a set of vertices $I \subseteq V$ with $|I| \geq K$
and $\{u, v\} \notin E$ for all $u, v \in I$ with $u \neq v$?

German: unabhängige Menge

Independent Set: Exercise (slido)



Does this graph have an independent set of size 3?



INDSET is NP-Complete (1)

Theorem (INDSET is NP-Complete)

INDSET *is NP-complete*.

INDSET is NP-Complete (1)

Theorem (INDSET is NP-Complete)

INDSET *is NP-complete*.

Proof.

INDSET $\in$ NP: guess and check.

...

INDSET is NP-Complete (2)

Proof (continued).

INDSET is NP-hard: We show $\text{CLIQUE} \leq_p \text{INDSET}$.

INDSET is NP-Complete (2)

Proof (continued).

INDSET is NP-hard: We show $\text{CLIQUE} \leq_p \text{INDSET}$.

We describe a polynomial reduction f .

Let $\langle G, K \rangle$ with $G = \langle V, E \rangle$ be the given input for CLIQUE.

INDSET is NP-Complete (2)

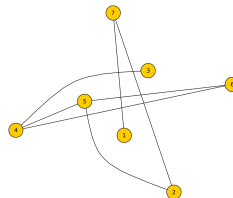
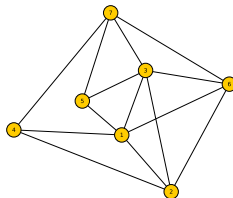
Proof (continued).

INDSET is NP-hard: We show $\text{CLIQUE} \leq_p \text{INDSET}$.

We describe a polynomial reduction f .

Let $\langle G, K \rangle$ with $G = \langle V, E \rangle$ be the given input for CLIQUE.

Then $f(\langle G, K \rangle)$ is the INDSET instance $\langle \overline{G}, K \rangle$, where $\overline{G} := \langle V, \overline{E} \rangle$ and $\overline{E} := \{\{u, v\} \subseteq V \mid u \neq v, \{u, v\} \notin E\}$. (This graph $\overline{G}$ is called the **complement graph** of G .)



INDSET is NP-Complete (2)

Proof (continued).

INDSET is NP-hard: We show $\text{CLIQUE} \leq_p \text{INDSET}$.

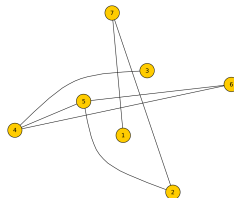
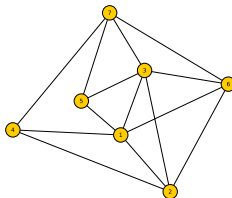
We describe a polynomial reduction f .

Let $\langle G, K \rangle$ with $G = \langle V, E \rangle$ be the given input for **CLIQUE**.

Then $f(\langle G, K \rangle)$ is the **INDSET** instance $\langle \overline{G}, K \rangle$, where $\overline{G} := \langle V, \overline{E} \rangle$ and $\overline{E} := \{\{u, v\} \subseteq V \mid u \neq v, \{u, v\} \notin E\}$.
(This graph $\overline{G}$ is called the **complement graph** of G .)

Clearly f can be computed in polynomial time.

...



INDSET is NP-Complete (3)

Proof (continued).

We have:

$$\langle \langle V, E \rangle, K \rangle \in \text{CLIQUE}$$

iff there exists a set $V' \subseteq V$ with $|V'| \geq K$
and $\{u, v\} \in E$ for all $u, v \in V'$ with $u \neq v$

iff there exists a set $V' \subseteq V$ with $|V'| \geq K$
and $\{u, v\} \notin \overline{E}$ for all $u, v \in V'$ with $u \neq v$

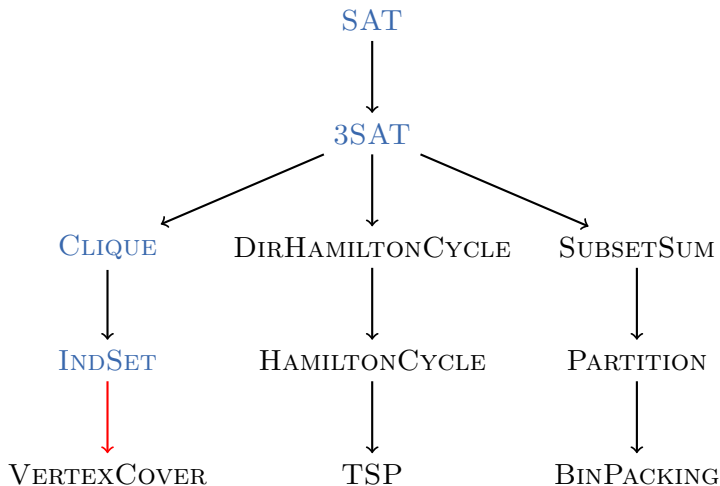
iff $\langle \langle V, \overline{E} \rangle, K \rangle \in \text{INDSET}$

iff $f(\langle \langle V, E \rangle, K \rangle) \in \text{INDSET}$

and hence f is a reduction.



$\text{INDSET} \leq_p \text{VERTEXCOVER}$



VERTEXCOVER

Definition (VERTEXCOVER)

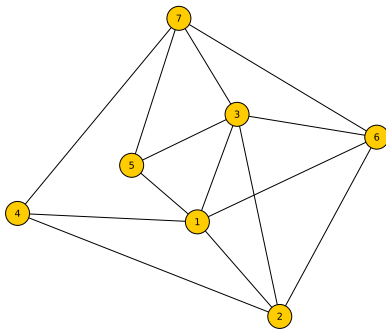
The problem **VERTEXCOVER** is defined as follows:

Given: undirected graph $G = \langle V, E \rangle$, number $K \in \mathbb{N}_0$

Question: Does G have a vertex cover of size at most K ,
i. e., a set of vertices $C \subseteq V$ with $|C| \leq K$ and $\{u, v\} \cap C \neq \emptyset$
for all $\{u, v\} \in E$?

German: Knotenüberdeckung

Vertex Cover: Exercise (slido)



Does this graph have a vertex cover of size 4?



VERTEXCOVER is NP-Complete (1)

Theorem (VERTEXCOVER is NP-Complete)

VERTEXCOVER *is NP-complete.*

VERTEXCOVER is NP-Complete (2)

Proof.

VERTEXCOVER $\in$ NP: guess and check.

VERTEXCOVER is NP-Complete (2)

Proof.

VERTEXCOVER $\in$ NP: guess and check.

VERTEXCOVER is NP-hard:

We show $\text{INDSET} \leq_p \text{VERTEXCOVER}$.

VERTEXCOVER is NP-Complete (2)

Proof.

VERTEXCOVER $\in$ NP: guess and check.

VERTEXCOVER is NP-hard:

We show $\text{INDSET} \leq_p \text{VERTEXCOVER}$.

We describe a polynomial reduction f .

Let $\langle G, K \rangle$ with $G = \langle V, E \rangle$ be the given input for INDSET.

VERTEXCOVER is NP-Complete (2)

Proof.

VERTEXCOVER $\in$ NP: guess and check.

VERTEXCOVER is NP-hard:

We show $\text{INDSET} \leq_p \text{VERTEXCOVER}$.

We describe a polynomial reduction f .

Let $\langle G, K \rangle$ with $G = \langle V, E \rangle$ be the given input for INDSET.

Then $f(\langle G, K \rangle) := \langle G, |V| - K \rangle$.

This can clearly be computed in polynomial time.

...

VERTEXCOVER is NP-Complete (3)

Proof (continued).

For vertex set $V' \subseteq V$, we write $\overline{V'}$ for its **complement** $V \setminus V'$.

VERTEXCOVER is NP-Complete (3)

Proof (continued).

For vertex set $V' \subseteq V$, we write $\overline{V'}$ for its **complement** $V \setminus V'$.

Observation: a set of vertices is a vertex cover
iff its complement is an independent set.

VERTEXCOVER is NP-Complete (3)

Proof (continued).

For vertex set $V' \subseteq V$, we write $\overline{V'}$ for its **complement** $V \setminus V'$.

Observation: a set of vertices is a vertex cover
iff its complement is an independent set.

We thus have:

$$\begin{aligned}
 & \langle \langle V, E \rangle, K \rangle \in \text{INDSET} \\
 \text{iff} & \quad \langle V, E \rangle \text{ has an independent set } I \text{ with } |I| \geq K \\
 \text{iff} & \quad \langle V, E \rangle \text{ has a vertex cover } C \text{ with } |\overline{C}| \geq K \\
 \text{iff} & \quad \langle V, E \rangle \text{ has a vertex cover } C \text{ with } |C| \leq |V| - K \\
 \text{iff} & \quad \langle \langle V, E \rangle, |V| - K \rangle \in \text{VERTEXCOVER} \\
 \text{iff} & \quad f(\langle \langle V, E \rangle, K \rangle) \in \text{VERTEXCOVER}
 \end{aligned}$$

and hence f is a reduction.



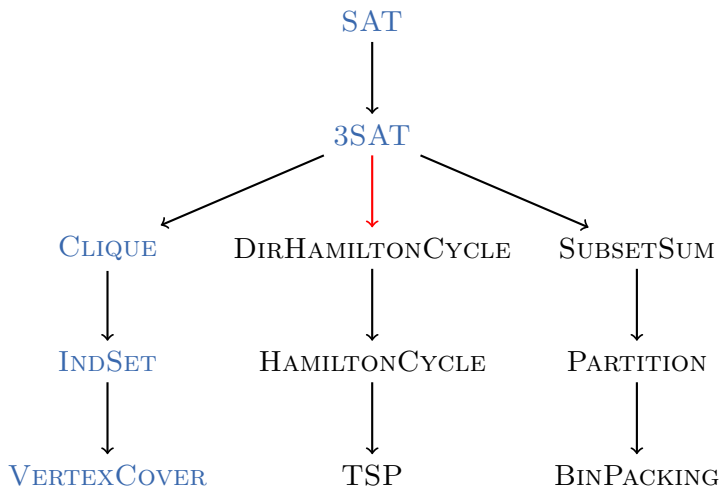
Questions



Questions?

Routing Problems

$3\text{SAT} \leq_p \text{DirHAMILTONCYCLE}$



DIRHAMILTONCYCLE is NP-Complete (1)

Definition (Reminder: DIRHAMILTONCYCLE)

The problem **DIRHAMILTONCYCLE** is defined as follows:

Given: directed graph $G = \langle V, E \rangle$

Question: Does G contain a Hamilton cycle?

DIRHAMILTONCYCLE is NP-Complete (1)

Definition (Reminder: DIRHAMILTONCYCLE)

The problem **DIRHAMILTONCYCLE** is defined as follows:

Given: directed graph $G = \langle V, E \rangle$

Question: Does G contain a Hamilton cycle?

Theorem

DIRHAMILTONCYCLE is NP-complete.

DIRHAMILTONCYCLE is NP-Complete (2)

Proof.

DIRHAMILTONCYCLE $\in$ NP: guess and check.

DIRHAMILTONCYCLE is NP-Complete (2)

Proof.

DIRHAMILTONCYCLE $\in$ NP: guess and check.

DIRHAMILTONCYCLE is NP-hard:

We show $3\text{SAT} \leq_p \text{DIRHAMILTONCYCLE}$.

DIRHAMILTONCYCLE is NP-Complete (2)

Proof.

DIRHAMILTONCYCLE $\in$ NP: guess and check.

DIRHAMILTONCYCLE is NP-hard:

We show $3SAT \leq_p$ DIRHAMILTONCYCLE.

- We are given a 3-CNF formula φ where each clause contains exactly three literals and no clause contains duplicated literals.
- We must, in polynomial time, construct a directed graph $G = \langle V, E \rangle$ such that:
 G contains a Hamilton cycle iff φ is satisfiable.
- construction of $\langle V, E \rangle$ on the following slides

DIRHAMILTONCYCLE is NP-Complete (3)

Proof (continued).

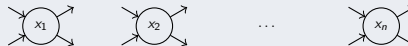
- Let $X_1, \dots, X_n$ be the atomic propositions in φ .
- Let $c_1, \dots, c_m$ be the clauses of φ with $c_i = (\ell_{i1} \vee \ell_{i2} \vee \ell_{i3})$.
- Construct a graph with $6m + n$ vertices (described on the following slides).

...

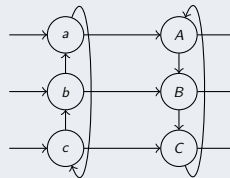
DIRHAMILTONCYCLE is NP-Complete (4)

Proof (continued).

- For every variable X_i , add vertex x_i with 2 incoming and 2 outgoing edges:



- For every clause c_j , add the subgraph C_j with 6 vertices:



- We describe later how to connect these parts.

DIRHAMILTONCYCLE is NP-Complete (5)

Proof (continued).

Let π be a Hamilton cycle of the total graph.

- Whenever π enters subgraph C_j from one of its “entrances”, it must leave via the corresponding “exit”:
($a \longrightarrow A$, $b \longrightarrow B$, $c \longrightarrow C$).
Otherwise, π cannot be a Hamilton cycle.
- Hamilton cycles can behave in the following ways with regard to C_j :
 - π passes through C_j once (from any entrance)
 - π passes through C_j twice (from any two entrances)
 - π passes through C_j three times (once from every entrance)

...

DIRHAMILTONCYCLE is NP-Complete (6)

Proof (continued).

Connect the “open ends” in the graph as follows:

- Identify entrances/exits of the clause subgraph C_j with the three literals in clause c_j .
- One exit of x_i is **positive**, the other one is **negative**.
- For the **positive** exit, determine the clauses in which the positive literal X_i occurs:
 - Connect the positive exit of x_i with the X_i -entrance of the first such clause graph.
 - Connect the X_i -exit of this clause graph with the X_i -entrance of the second such clause graph, and so on.
 - Connect the X_i -exit of the last such clause graph with the positive entrance of x_{i+1} (or x_1 if $i = n$).
- analogously for the **negative** exit of x_i and the literal $\neg X_i$

DIRHAMILTONCYCLE is NP-Complete (7)

Proof (continued).

The construction is polynomial and is a reduction:

($\Rightarrow$): **construct a Hamilton cycle from a satisfying assignment**

- Given a satisfying assignment $\mathcal{I}$, construct a Hamilton cycle that leaves x_i through the positive exit if $\mathcal{I}(X_i)$ is true and by the negative exit if $\mathcal{I}(X_i)$ is false.
- Afterwards, we visit all C_j -subgraphs for clauses that are satisfied by this literal.
- In total, we visit each C_j -subgraph 1–3 times.

...

DIRHAMILTONCYCLE is NP-Complete (8)

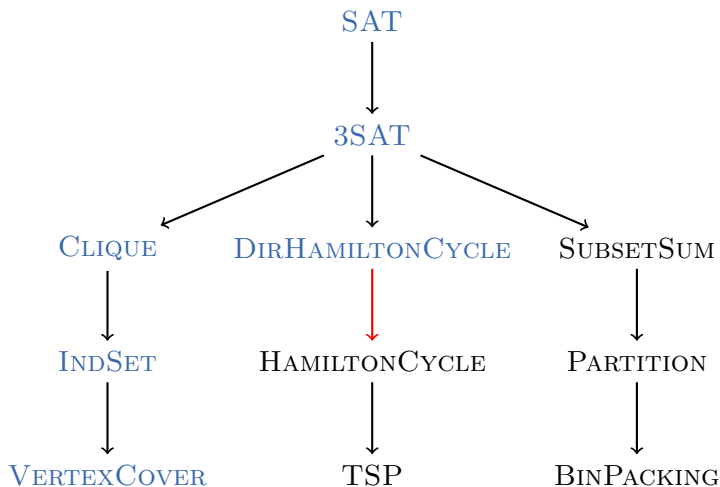
Proof (continued).

($\Leftarrow$): **construct a satisfying assignment from a Hamilton cycle**

- A Hamilton cycle visits every vertex x_i and leaves it by the positive or negative exit.
- Map X_i to true or false depending on which exit is used to leave x_i .
- Because the cycle must traverse each C_j -subgraph at least once (otherwise it is not a Hamilton cycle), this results in a satisfying assignment. (Details omitted.)



$\text{DirHAMILTONCYCLE} \leq_p \text{HAMILTONCYCLE}$



HAMILTONCYCLE is NP-Complete (1)

Definition (Reminder: HAMILTONCYCLE)

The problem **HAMILTONCYCLE** is defined as follows:

Given: undirected graph $G = \langle V, E \rangle$

Question: Does G contain a Hamilton cycle?

HAMILTONCYCLE is NP-Complete (1)

Definition (Reminder: HAMILTONCYCLE)

The problem **HAMILTONCYCLE** is defined as follows:

Given: undirected graph $G = \langle V, E \rangle$

Question: Does G contain a Hamilton cycle?

Theorem

HAMILTONCYCLE *is NP-complete.*

HAMILTONCYCLE is NP-Complete (2)

Proof sketch.

HAMILTONCYCLE $\in$ NP: guess and check.

HAMILTONCYCLE is NP-Complete (2)

Proof sketch.

HAMILTONCYCLE $\in$ NP: guess and check.

HAMILTONCYCLE is NP-hard: We show
 $\text{DIRHAMILTONCYCLE} \leq_p \text{HAMILTONCYCLE}$.

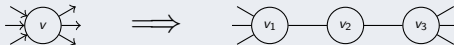
HAMILTONCYCLE is NP-Complete (2)

Proof sketch.

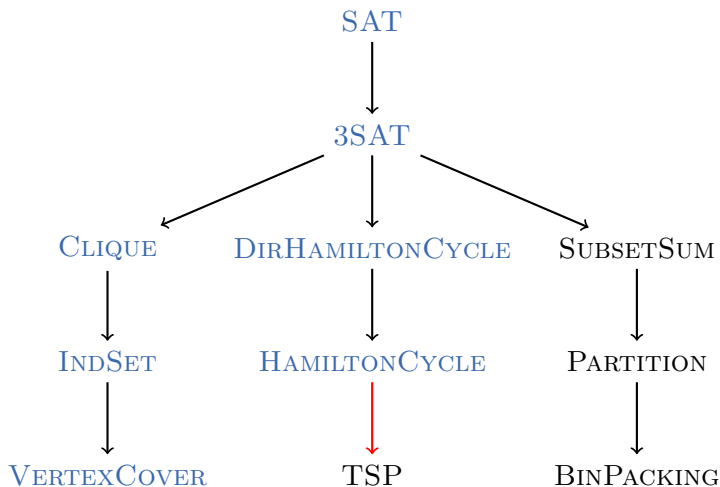
HAMILTONCYCLE $\in$ NP: guess and check.

HAMILTONCYCLE is NP-hard: We show
 DIRHAMILTONCYCLE $\leq_p$ HAMILTONCYCLE.

Basic building block of the reduction:



HAMILTONCYCLE $\leq_p$ TSP



TSP is NP-Complete (1)

Definition (Reminder: TSP)

TSP (traveling salesperson problem) is the following decision problem:

- **Given:** finite set $S \neq \emptyset$ of cities, symmetric cost function $\text{cost} : S \times S \rightarrow \mathbb{N}_0$, cost bound $K \in \mathbb{N}_0$
- **Question:** Is there a tour with total cost at most K , i. e., a permutation $\langle s_1, \dots, s_n \rangle$ of the cities with
$$\sum_{i=1}^{n-1} \text{cost}(s_i, s_{i+1}) + \text{cost}(s_n, s_1) \leq K?$$

German: Problem der/des Handlungsreisenden

Theorem

TSP is NP-complete.

TSP is NP-Complete (2)

Proof.

TSP $\in$ NP: guess and check.

TSP is NP-hard: We showed $\text{HAMILTONCYCLE} \leq_p \text{TSP}$
in Chapter D2. □

Questions



Questions?

Summary

Summary

- In this chapter we showed NP-completeness of
 - three classical graph problems:
CLIQUE, INDSET, VERTEXCOVER
 - three classical routing problems:
DIRHAMILTONCYCLE, HAMILTONCYCLE, TSP