# Theory of Computer Science

## E4. Some NP-Complete Problems, Part I

Malte Helmert

University of Basel

May 29, 2017

# Theory of Computer Science

May 29, 2017 — E4. Some NP-Complete Problems, Part I

E4.1 Overview

E4.2 3SAT

E4.3 Graph Problems

E4.4 Summary

# Overview: Course

contents of this course:

- logic ✓
  - ▷ How can knowledge be represented? How can reasoning be automated?
- automata theory and formal languages ✓
  - ▷ What is a computation?
- computability theory ✓
  - ▷ What can be computed at all?
- complexity theory
  - ▷ What can be computed efficiently?

# Overview: Complexity Theory

Complexity Theory

E1. Motivation and Introduction
E2. P, NP and Polynomial Reductions
E3. Cook-Levin Theorem
E4. Some NP-Complete Problems, Part I
E5. Some NP-Complete Problems, Part II

# Further Reading (German)

### Literature for this Chapter (German)

Theoretische Informatik – kurz gefasst
by Uwe Schöning (5th edition)

- Chapter 3.3

# Further Reading (English)

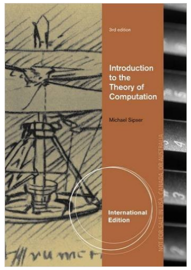
### Literature for this Chapter (English)

Introduction to the Theory of Computation
by Michael Sipser (3rd edition)

- Chapter 7.4 and 7.5

Note:

- Sipser does not cover all problems that we do.



# E4.1 Overview

# Proving NP-Completeness by Reduction

- The proof of NP-completeness for $\mathrm{SAT}$ was complicated.
- But with its help, we can now prove much more easily that further problems are NP-complete.

### Theorem (Proving NP-Completeness by Reduction)

*Let $A$ and $B$ be problems such that:*

- *$A$ is NP-hard, and*
- *$A \leq_{\mathrm{p}} B$.*

*Then $B$ is also NP-hard.*
*If furthermore $B \in \mathrm{NP}$, then $B$ is NP-complete.*

## Proving NP-Completeness by Reduction: Proof

Proof.

First part: We must show $X \leq_\mathrm{p} B$ for all $X \in \mathrm{NP}$.

From $X \leq_\mathrm{p} A$ (because $A$ is NP-hard) and $A \leq_\mathrm{p} B$ (by prerequisite), this follows due to the transitivity of $\leq_\mathrm{p}$.

Second part: follows directly by definition of NP-completeness. □

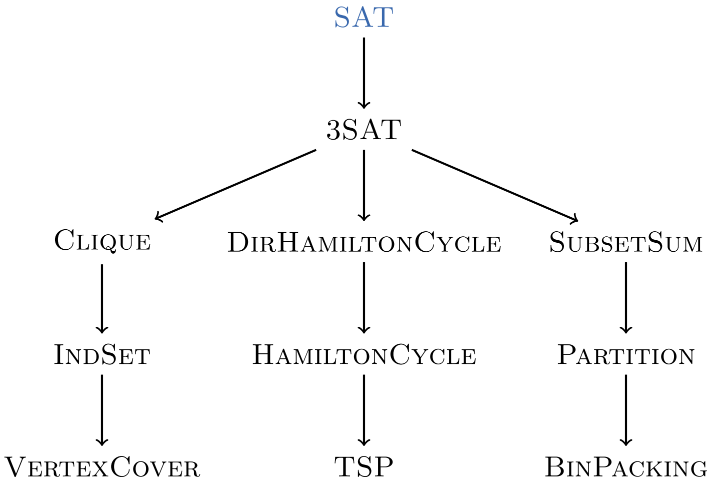
## NP-Complete Problems

- There are thousands of known NP-complete problems.
- An extensive catalog of NP-complete problems from many areas of computer science is contained in:

  *Michael R. Garey and David S. Johnson:*
  *Computers and Intractability —*
  *A Guide to the Theory of NP-Completeness*
  *W. H. Freeman, 1979.*

- In the remaining two chapters, we get to know some of these problems.

## Overview of the Reductions



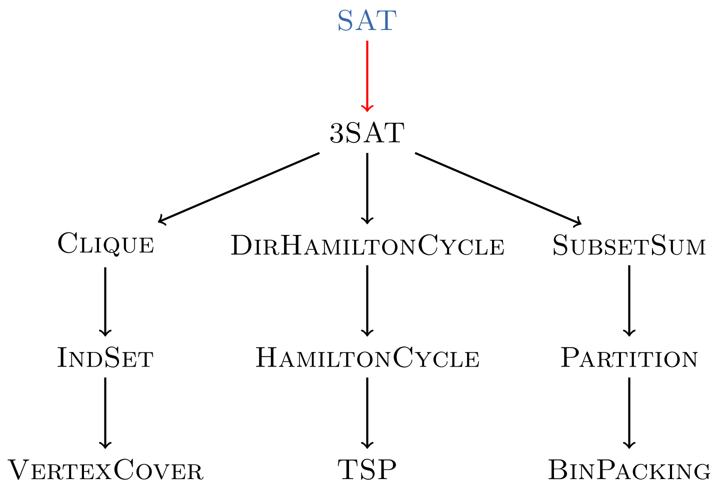
## What Do We Have to Do?

- We want to show the NP-completeness of these 11 problems.
- We already know that SAT is NP-complete.
- Hence it is sufficient to show
  - that polynomial reductions exist for all edges in the figure (and thus all problems are NP-hard)
  - and that the problems are all in NP.

(It would be sufficient to show membership in NP only for the leaves in the figure. But membership is so easy to show that this would not save any work.)

# E4.2 3SAT

## SAT $\leq_p$ 3SAT



## SAT and 3SAT

### Definition (Reminder: SAT)

The problem SAT (satisfiability) is defined as follows:

Given: a propositional logic formula $\varphi$

Question: Is $\varphi$ satisfiable?

### Definition (3SAT)

The problem 3SAT is defined as follows:

Given: a propositional logic formula $\varphi$ in conjunctive normal form with at most three literals per clause

Question: Is $\varphi$ satisfiable?

## 3SAT is NP-Complete (1)

### Theorem (3SAT is NP-Complete)

3SAT *is NP-complete.*

## 3SAT is NP-Complete (2)

Proof.

3SAT $\in$ NP: guess and check.

3SAT is NP-hard: We show SAT $\leq_p$ 3SAT.

- Let $\varphi$ be the given input for SAT. Let $Sub(\varphi)$ denote the set of subformulas of $\varphi$, including $\varphi$ itself.
- For all $\psi \in Sub(\varphi)$, we introduce a new proposition $X_\psi$.
- For each new proposition $X_\psi$, define the following auxiliary formula $\chi_\psi$:
  - If $\psi = A$ for an atom $A$: $\chi_\psi = (X_\psi \leftrightarrow A)$
  - If $\psi = \neg\psi'$: $\chi_\psi = (X_\psi \leftrightarrow \neg X_{\psi'})$
  - If $\psi = (\psi' \land \psi'')$: $\chi_\psi = (X_\psi \leftrightarrow (X_{\psi'} \land X_{\psi''}))$
  - If $\psi = (\psi' \lor \psi'')$: $\chi_\psi = (X_\psi \leftrightarrow (X_{\psi'} \lor X_{\psi''}))$

...

## 3SAT is NP-Complete (3)

Proof (continued).

- Consider the conjunction of all these auxiliary formulas, $\chi_{\text{all}} := \bigwedge_{\psi \in Sub(\varphi)} \chi_\psi$.
- Every interpretation $\mathcal{I}$ of the original variables can be extended to a model $\mathcal{I}'$ of $\chi_{\text{all}}$ in exactly one way: for each $\psi \in Sub(\varphi)$, set $\mathcal{I}'(X_\psi) = 1$ iff $\mathcal{I} \models \psi$.
- It follows that $\varphi$ is satisfiable iff $(\chi_{\text{all}} \land X_\varphi)$ is satisfiable.
- This formula can be computed in linear time.
- It can also be converted to 3-CNF in linear time because it is the conjunction of constant-size parts involving at most three variables each. (Each part can be converted to 3-CNF independently.)
- Hence, this describes a polynomial-time reduction.

□

## Restricted 3SAT

Note: 3SAT remains NP-complete if we also require that

- every clause contains exactly three literals and
- a clause may not contain the same literal twice

Idea:

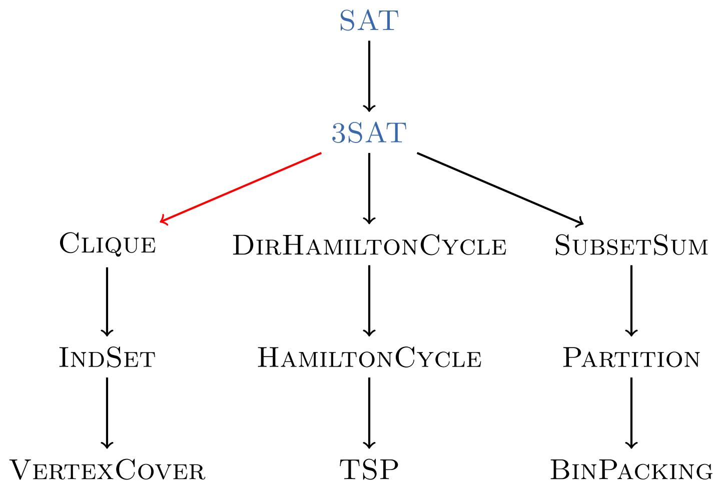
- remove duplicated literals from each clause.
- add new variables: $X$, $Y$, $Z$
- add new clauses: $(X \lor Y \lor Z)$, $(X \lor Y \lor \neg Z)$, $(X \lor \neg Y \lor Z)$, $(\neg X \lor Y \lor Z)$, $(X \lor \neg Y \lor \neg Z)$, $(\neg X \lor Y \lor \neg Z)$, $(\neg X \lor \neg Y \lor Z)$

$\rightsquigarrow$ satisfied if and only if $X$, $Y$, $Z$ are all true

- fill up clauses with fewer than three literals with $\neg X$ and if necessary additionally with $\neg Y$

# E4.3 Graph Problems

## $3\mathrm{SAT} \le_p$ 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

## 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-hard: We show $3\mathrm{SAT} \le_p$ Clique.

- We are given a 3-CNF formula $\varphi$, 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 $\varphi$ is satisfiable.

$\rightsquigarrow$ construction of $V, E, K$ on the following slides.

...

## Clique is NP-Complete (3)

Proof (continued).

Let $m$ be the number of clauses in $\varphi$.

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 $l_{i'j'}$ are not complementary literals
- $K = m$

$\rightsquigarrow$ obviously polynomially computable

to show: reduction property ...

## Clique is NP-Complete (4)

Proof (continued).

($\Rightarrow$): If $\varphi$ 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 $\varphi$ 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).

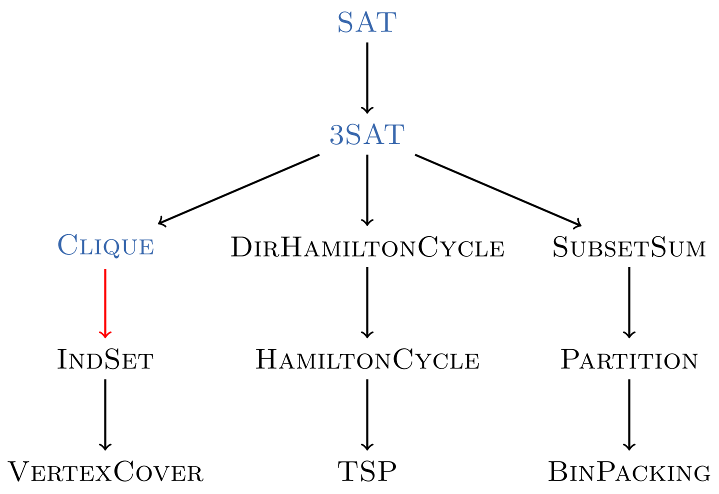
$\rightsquigarrow$ exactly one vertex per clause is included in $C$

- 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 1 (true).
- If a vertex corresp. to $\neg X$ was chosen, map $X$ to 0 (false).
- If neither was chosen, arbitrarily map $X$ to 0 or 1.

$\rightsquigarrow$ satisfying assignment

□

## Clique $\leq_p$ IndSet



SAT

3SAT

Clique

DirHamiltonCycle

SubsetSum

IndSet

HamiltonCycle

Partition

VertexCover

TSP

BinPacking

## 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

## IndSet is NP-Complete (1)

Theorem (IndSet is NP-Complete)

IndSet *is NP-complete.*

## IndSet is NP-Complete (2)

Proof.

IndSet $\in$ NP: guess and check.

IndSet is NP-hard: We show Clique $\leq_p$ 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)

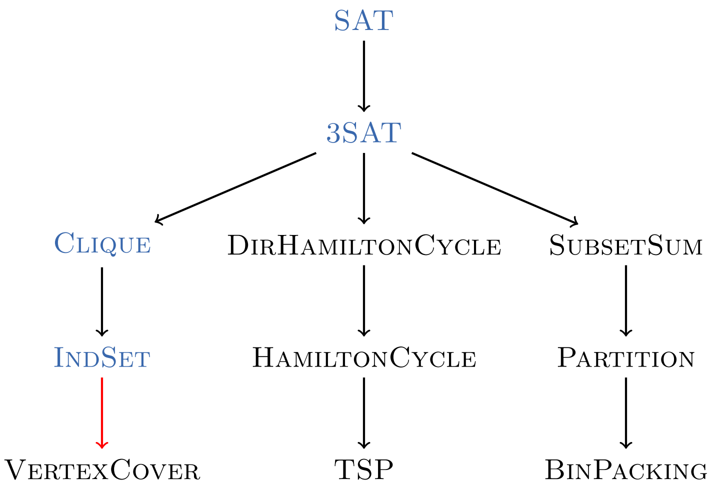
Proof (continued).

We have:

$$\begin{array}{ll}
& \langle\langle V, E \rangle, K \rangle \in \textsc{Clique} \\
\text{iff} & \text{there exists a set } V' \subseteq V \text{ with } |V'| \geq K \\
& \quad \text{and } \{u, v\} \in E \text{ for all } u, v \in V' \text{ with } u \neq v \\
\text{iff} & \text{there exists a set } V' \subseteq V \text{ with } |V'| \geq K \\
& \quad \text{and } \{u, v\} \notin \overline{E} \text{ for all } u, v \in V' \text{ with } u \neq v \\
\text{iff} & \langle\langle V, \overline{E} \rangle, K \rangle \in \textsc{IndSet} \\
\text{iff} & f(\langle\langle V, E \rangle, K \rangle) \in \textsc{IndSet}
\end{array}$$

and hence $f$ is a reduction. □

## IndSet $\leq_p$ VertexCover



SAT

3SAT

Clique DirHamiltonCycle SubsetSum

IndSet HamiltonCycle Partition

VertexCover TSP BinPacking

## 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

## 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-hard:
We show IndSet $\leq_p$ 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 $V \setminus V'$.

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

We thus have:

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

and hence $f$ is a reduction. □

# E4.4 Summary

## Summary

- Thousands of important problems are NP-complete.
- Usually, the easiest way to show that a problem is NP-complete is to
  - show that it is in NP with a guess-and-check algorithm, and
  - polynomially reduce a known NP-complete to it.
- In this chapter we showed NP-completeness of:
  - 3SAT, a restricted version of SAT
  - three classical graph problems: Clique, IndSet, VertexCover