

## Summary: Lecture 7

Summary for the chapters 9.1 and 9.2. [2, 1]

### NP-Completeness

#### NP

Class of languages decided by nondeterministic Turing machines in polynomial time.  
Most problems are in NP.

#### NP-completeness:

- easiest problems among those we do not know how to solve efficiently
- if  $P \neq NP$  can be proven: exact border of efficient solvability is found
- best bet for proving  $P=NP$ : show that some NP-complete problem is  $P$
- Until then, the NP-complete problems are the least likely ones in NP to be efficiently solved
- Where is the line between P and NP?

#### Language $L$

$L = \{x : (x, y) \in R \text{ for some } y\}$   
 $L$  gets an input  $x$  and finds a  $y$  with  $(x, y) \in R$  and the relation  $R \subseteq \Sigma^* \times \Sigma^*$ .

#### Polynomially decidable:

- $R$  is polynomially decidable if there is a deterministic Turing machine deciding the language  $L$  in polynomial time
- then the relation  $R$  (not the language  $L$ ) is polynomially decidable

#### Polynomially balanced:

- $R$  is polynomially balanced if  $(x, y) \in R$  implies  $|y| \leq |x|^k$  for some  $k \geq 1$   
→ length of the second component is bounded by a polynomial in the length of the first
- then the relation  $R$  (not the language  $L$ ) is polynomially balanced

#### NP

The language  $L \subseteq \Sigma^*$  is in NP only if there is a polynomially decidable and polynomially balanced relation  $R$  such that  $L = \{x : (x, y) \in R \text{ for some } y\}$ .

For example: Is there a satisfying assignment ( $y$ ) for a formula ( $x$ )?

Why is  $R \subseteq \Sigma^* \times \Sigma^*$ ? Is the input formula and the truth assignment  $\in \Sigma^*$ ?

#### Proof idea:

- 

TODO

proof

Questions:

## Succinct certificate (for NP-complete problems)

- *yes* instance of  $x$  has a polynomial witness  $y$  (certificate)
- *no* instances don't have such a certificate
- Examples:
  - SAT: certificate is the truth assignment
  - HAMILTONPATH: certificate is the hamilton path of a graph

## Typical problems in NP

- sometimes the optimum needs to be found
- sometimes any object that fits the specification is enough
- constraints can be added to optimization problems

## 3Sat is NP-complete

### SAT

The SAT (satisfiability) problem is the problem of determining if there exists an interpretation that satisfies a given Boolean formula. [4]

### 3SAT

Like the SAT problem, 3SAT is determining the satisfiability of a formula in CNF where each clause is limited to at most three literals.

- $k$ SAT with  $k \geq 1$  is a special case of SAT

## Reduction from SAT to 3SAT: [3]

- the reduction replaces each clause with a set of clauses, each having exactly three literals
- rewrite the clauses of the input
- example:

$$\begin{aligned} & (x_1) \wedge (x_1 \vee \bar{x}_2) \wedge (x_2 \vee x_3 \vee x_5) \wedge (x_1 \vee x_4 \vee \bar{x}_6 \vee \bar{x}_7) \wedge (x_1 \wedge x_2 \wedge \bar{x}_3 \vee x_5 \vee x_7) \\ & \equiv (x_1 \vee x_1 \vee x_1) \wedge (x_2 \vee x_3 \vee x_5) \end{aligned}$$

## 3Sat with more restrictions

3SAT remains NP-complete even for expressions in which each variable is restricted to appear at most three times and each literal at most twice.

## Proof idea:

- if a variable appears more often than twice:
  - introduce new variables and make sure (with the introduction of new clauses) that they have the same truthvalue as the original variable

TODO

bipartite graph or whatever ?

Questions:

## 2Sat in P (graph construction)

### 2-Sat

Like the SAT and 3-SAT problem, 2-SAT is determining the satisfiability of a formula in CNF where each clause is limited to at most two literals.

- let  $\psi$  be an instance of 2SAT (clauses with two literals each)
- construct formula  $\psi$  as graph
- the nodes are the variables (node for  $a$  and  $\neg a$ )
- for clauses  $(\neg a \vee b) \equiv a \rightarrow b$ : edge from the node  $a$  to the node  $b$
- paths in  $G$  are implications (implication is transitive)

TODO

Questions:

## 2Sat in NL

### Title

Content

TODO

Questions:

## MaxSat is NP-complete

### Title

Content

TODO

Questions:

## NaeSat is NP-complete

### Title

Content

TODO

Questions:

## References

- [1] Martin Berglund. *Lecture notes in Computational Complexity*.
- [2] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley Publishing Company, 1994.
- [3] Swagato Sanyal. *Reduction from SAT to 3SAT*. <https://cse.iitkgp.ac.in/~palash/2018AlgoDesignAnalysis/SAT-3SAT.pdf>, last opened: 29.11.22.
- [4] Prof. Dr. Thomas Schwentick. *Lecture notes in Grundbegriffe der theoretischen Informatik*. [https://www.cs.tu-dortmund.de/nps/de/Studium/Ordnungen\\_Handbuecher\\_Beschluesse/Modulhandbuecher/Archiv/Bachelor\\_LA\\_GyGe\\_Inf\\_Modellv/\\_Module/INF-BfP-GTI/index.html](https://www.cs.tu-dortmund.de/nps/de/Studium/Ordnungen_Handbuecher_Beschluesse/Modulhandbuecher/Archiv/Bachelor_LA_GyGe_Inf_Modellv/_Module/INF-BfP-GTI/index.html).