Cryptocomplexity II

Kryptokomplexität II

Sommersemester 2024

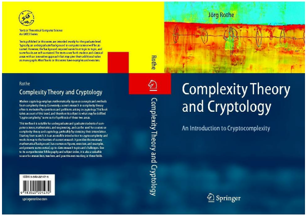
Chapter 7: Reminder: Some Foundations of Complexity Theory

Dozent: Prof. Dr. J. Rothe

Literature

Jörg Rothe: “Komplexitätstheorie und Kryptologie. Eine Einführung in Kryptokomplexität”, eXamen.Press, Springer-Verlag, 2008

Jörg Rothe: “Complexity Theory and Cryptology. An Introduction to Cryptocomplexity”, EATCS Texts in Theoretical Computer Science, Springer-Verlag, 2005

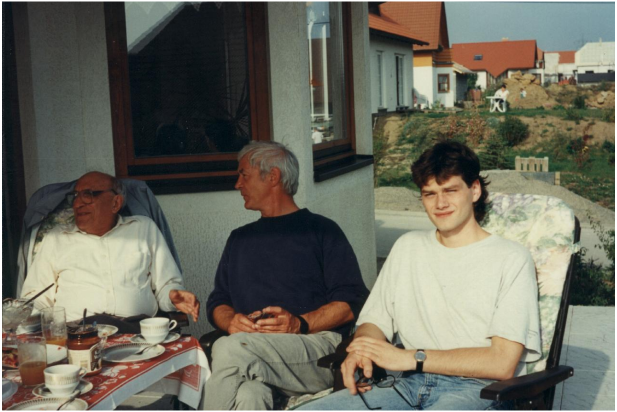


Literature

- **Gerd Wechsung:** **"Vorlesungen zur Komplexitätstheorie"**, Teubner-Verlag, Stuttgart, 2000

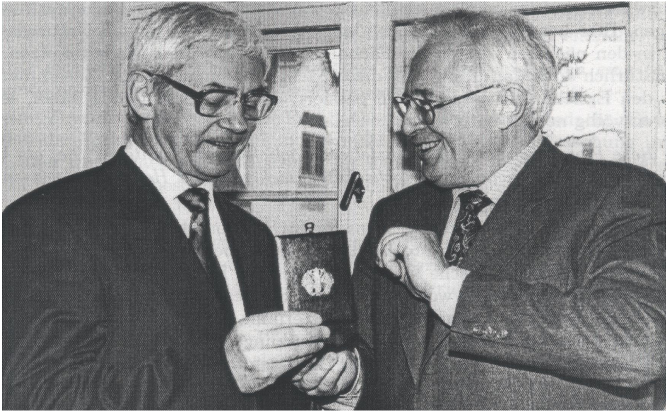
Literature

- **Gerd Wechsung: "Vorlesungen zur Komplexitätstheorie"**, Teubner-Verlag, Stuttgart, 2000



Literature

- **Gerd Wechsung: "Vorlesungen zur Komplexitätstheorie"**, Teubner-Verlag, Stuttgart, 2000



Literature

- **Gerd Wechsung:** **"Vorlesungen zur Komplexitätstheorie"**, Teubner-Verlag, Stuttgart, 2000

Groß-Demo gegen das SED-Regime heute vor 25 ... http://www.tlz.de/startseite/detail/-/specific/Gross...

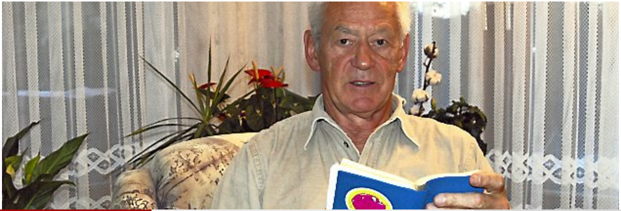
DIENSTAG, 04. NOVEMBER 2014

Thüringische Landeszeitung TLZ.DE

Groß-Demo gegen das SED-Regime heute vor 25 Jahren in Jena

04.11.2014 - 20:01 Uhr

Der Alt-Prorektor und DA-Mitgründer Gerd Wechsung erinnert sich.



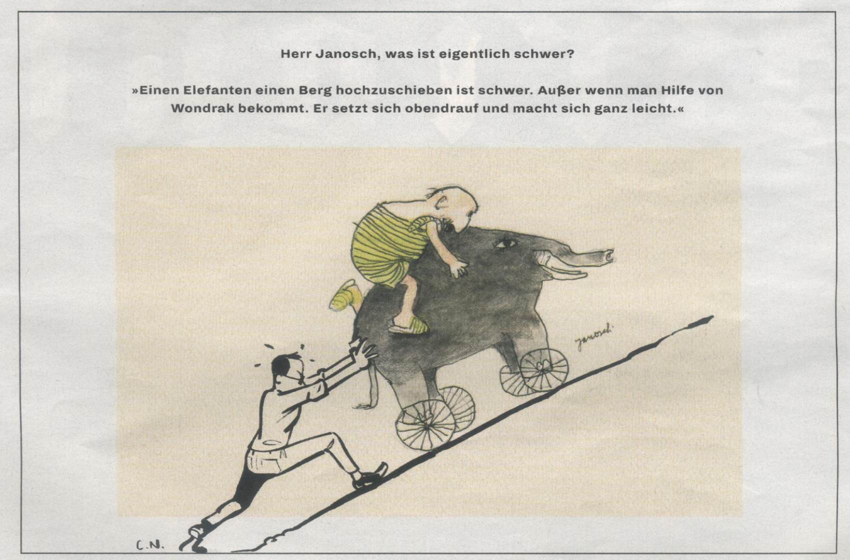
Literature

- **Gerd Wechsung: "Vorlesungen zur Komplexitätstheorie"**, Teubner-Verlag, Stuttgart, 2000
- **Lane A. Hemaspaandra and Mitsunori Ogihara: "The Complexity Theory Companion"**, EATCS Texts in Theoretical Computer Science, Springer-Verlag, Berlin, Heidelberg, New York, 2002
- **Christos Papadimitriou: "Computational Complexity"**, 2. Auflage, reprinted with corrections, Addison-Wesley, 1995
- **Danielo Bovet and Pierluigi Crescenzi: "Introduction to the Theory of Complexity"**, Prentice Hall, 1993

Literature

- **D. Du and K. Ko: "Theory of Computational Complexity"**, John Wiley and Sons, 2000
- **J. Balcázar, J. Díaz, and J. Gabarró: "Structural Complexity I + II"**, EATCS Monographs on Theoretical Computer Science, Berlin, Heidelberg, New York, vol. I (1995, 2. Auflage), vol. II (1990)
- **Ingo Wegener: "Komplexitätstheorie. Grenzen der Effizienz von Algorithmen"**, Springer-Verlag, Berlin, Heidelberg, New York, 2003
- **Klaus W. Wagner and Gerd Wechsung: "Computational Complexity"**, D. Reidel Publishing Company, 1986

Actually, What Does It Mean to Be a “Hard” Problem?



Herr Janosch, was ist eigentlich schwer?

»Einen Elefanten einen Berg hochzuschieben ist schwer. Außer wenn man Hilfe von Wondrak bekommt. Er setzt sich obendrauf und macht sich ganz leicht.«

Another Intractable Problem

How "Hard" Is $S = \{x2^{|x|}x \mid x \in \{0,1\}^*\}$?

1. Turing machines with one read-only input tape and one read-write working tape can solve S in **real-time**, i.e., the number of steps in the computation equals the length of the input.
2. Turing machines with only one working tape and no separate input tape require **time at least quadratic** in the input size to solve S.
3. Alternating Turing machines need **time no more than logarithmic** in the input size to solve S.
4. **Finite automata cannot solve S at all.**

   Note that finite automata can be considered to be very restricted Turing machines, which are equipped only with a one-way read-only input tape (i.e., the head is allowed to go only from left to right in each step), have no working tape, and must finish their work in real-time.

A Problem's Complexity is Determined by:

- the *computational model* (or *algorithmic device*) used—e.g., the two-way, multitape Turing machine;
- the *computational paradigm* (or *acceptance mode*) of this computational model—e.g., Turing machines are
  - deterministic or
  - nondeterministic or
  - probabilistic or
  - alternating or etc.
- the *complexity measure* (or *resource*) used—e.g.,
  - the time (the number of steps executed in the computation) or
  - the space (the number of tape cells used in the computation) or etc.

  needed to solve the problem (in either the *worst-case* or the *average-case complexity model*).

Where Do the Problems Come From?

Complexity theory studies important, interesting, natural problems from almost every field of sciences, including areas as diverse as

- logic,
- graph theory,
- algebra and number theory,
- algorithmics,
- cryptography,
- coding and information theory,
- data compression,
- the theory of formal languages and automata,
- circuit theory,
- genome sequencing,
- social choice theory, and many more.

Where Do the Problems Come From? Examples

- The *satisfiability problem* of propositional logic:

$$\text{SAT} = \left\{ \varphi \;\middle|\; \varphi \text{ is a satisfiable boolean formula} \right\}$$

- The *clique problem* of graph theory:

$$\text{CLIQUE} = \{(G,k) \mid G \text{ is a graph that has a clique of size } \geq k\}$$

- The *primality problem* and the *quadratic residue problem* of algebra and number theory:

$$\text{PRIMES} = \{\mathtt{bin}(n) \mid n \text{ is a prime number}\}$$

$$\text{QR} = \left\{ (x,n) \;\middle|\; \begin{array}{l} x \in \mathbb{Z}_n^* \text{ and } n \in \mathbb{N} \text{ are encoded in binary} \\ \text{and } x \text{ is a quadratic residue mod } n \end{array} \right\}$$

Where Do the Problems Come From? Examples

- The *multiprocessor job scheduling problem* in algorithmics:
  "Given a list $J = (j_1, j_2, \ldots, j_k)$ of jobs, j_i having length ℓ_i, m processors, and a bound t, is it possible to schedule all jobs in J on the processors such that none overlap and the total time to process all jobs is at most t?"

  Decision problems like this have related optimization problems.

- The (functional) *problem of breaking RSA* in cryptography:
  "Given the public RSA key (n, e) in binary notation, determine the corresponding private key d."

- How does this relate to the *factoring problem*: "Given a number n in binary notation, determine its prime factors"?

Where Do the Problems Come From? Examples

- From the theory of formal languages and automata:
  - The *halting problem for Turing machines*:
    "Given a Turing machine M and an input x, does $M(x)$ ever halt?"
  - The *equivalence problem for context-free grammars*:
    "Given two context-free grammars, G_1 and G_2, are they equivalent (i.e., does it hold that $L(G_1) = L(G_2)$)?"
- From social choice theory:
  - The *winner problem for plurality elections*:
    "Given an election (C, V) and a distinguished candidate $c \in C$, is c a plurality winner of (C, V)?"
  - The *(coalitional weighted) manipulation problem*:
    "Given a candidate set C, a candidate $c \in C$, the votes and weights of the nonmanipulative voters, and the weights of the manipulators, can the manipulators cast their votes so that c wins?"

Tasks and Aims of Complexity Theory

- ***Classify*** problems in terms of their intrinsic complexity:
  - Prove an (algorithmic) upper bound for the problem;
  - Prove a *lower bound* for the problem.
- ***Compare*** problems according to their computational complexity via *complexity-bounded reducibilities*.
- ***Determine*** the “hardest” problems of complexity classes in terms of completeness w.r.t. some reducibility.
- ***Prove*** structural properties of complexity classes and hierarchies.

Reminder: Some Central Complexity Classes

Space classes		
L	=	DSPACE(log)
NL	=	NSPACE(log)
LINSPACE	=	DSPACE($\mathbb{L}$in)
NLINSPACE	=	NSPACE($\mathbb{L}$in)
PSPACE	=	DSPACE($\mathbb{P}$ol)
NPSPACE	=	NSPACE($\mathbb{P}$ol)
EXPSPACE	=	DSPACE($2^{\mathbb{P}\text{ol}}$)
NEXPSPACE	=	NSPACE($2^{\mathbb{P}\text{ol}}$)

Reminder: Some Central Complexity Classes

Time classes		
REALTIME	$=$	$\mathrm{DTIME}(\mathrm{id})$
LINTIME	$=$	$\mathrm{DTIME}(\mathbb{L}\mathrm{in})$
P	$=$	$\mathrm{DTIME}(\mathbb{P}\mathrm{ol})$
NP	$=$	$\mathrm{NTIME}(\mathbb{P}\mathrm{ol})$
E	$=$	$\mathrm{DTIME}(2^{\mathbb{L}\mathrm{in}})$
NE	$=$	$\mathrm{NTIME}(2^{\mathbb{L}\mathrm{in}})$
EXP	$=$	$\mathrm{DTIME}(2^{\mathbb{P}\mathrm{ol}})$
NEXP	$=$	$\mathrm{NTIME}(2^{\mathbb{P}\mathrm{ol}})$

Reminder: Simple Inclusions

Theorem

$$\mathrm{L} \subseteq \mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}.$$

Reminder: Many-One Reducibility and Completeness

Definition

- Let $\Sigma = \{0, 1\}$ be a fixed alphabet, and let $A, B \subseteq \Sigma^*$.
- Let FP denote the set of polynomial-time computable functions mapping from Σ^* to Σ^*.
- Let $\mathcal{C}$ be any complexity class.

1. Define the *polynomial-time many-one reducibility*, denoted by $\leq_{\mathrm{m}}^{\mathrm{p}}$, as follows: $A \leq_{\mathrm{m}}^{\mathrm{p}} B$ if there is a function $f \in \mathrm{FP}$ such that

$$(\forall x \in \Sigma^*)\,[x \in A \iff f(x) \in B].$$

Reminder: Many-One Reducibility and Completeness

Definition (continued)

2. A set B is *$\leq_{\mathrm{m}}^{\mathrm{p}}$-hard for $\mathcal{C}$* if $A \leq_{\mathrm{m}}^{\mathrm{p}} B$ for each $A \in \mathcal{C}$.

3. A set B is *$\leq_{\mathrm{m}}^{\mathrm{p}}$-complete for $\mathcal{C}$* if
   1. B is $\leq_{\mathrm{m}}^{\mathrm{p}}$-hard for $\mathcal{C}$ (lower bound) and
   2. $B \in \mathcal{C}$ (upper bound).

4. $\mathcal{C}$ is said to be *closed under the $\leq_{\mathrm{m}}^{\mathrm{p}}$-reducibility* ($\leq_{\mathrm{m}}^{\mathrm{p}}$*-closed*, for short) if for any two sets A and B,

$$\text{if } A \leq_{\mathrm{m}}^{\mathrm{p}} B \text{ and } B \in \mathcal{C}, \text{ then } A \in \mathcal{C}.$$

Reminder: Properties of $\leq_{m}^{p}$

Lemma

1. *$A \leq_{m}^{p} B$ implies $\overline{A} \leq_{m}^{p} \overline{B}$, yet in general it is not true that $A \leq_{m}^{p} \overline{A}$.*

2. *The relation $\leq_{m}^{p}$ is both reflexive and transitive, yet not antisymmetric.*

3. P, NP, *and* PSPACE *are $\leq_{m}^{p}$-closed.*
   That is, upper bounds are inherited downward with respect to $\leq_{m}^{p}$.

4. *If $A \leq_{m}^{p} B$ and A is $\leq_{m}^{p}$-hard for some complexity class $\mathcal{C}$, then B is $\leq_{m}^{p}$-hard for $\mathcal{C}$.*
   That is, lower bounds are inherited upward with respect to $\leq_{m}^{p}$.

Reminder: Properties of $\leq_{\mathrm{m}}^{\mathrm{p}}$

Lemma (continued)

5. *Let $\mathcal{C}$ and $\mathcal{D}$ be any complexity classes. If $\mathcal{C}$ is $\leq_{\mathrm{m}}^{\mathrm{p}}$-closed and B is $\leq_{\mathrm{m}}^{\mathrm{p}}$-complete for $\mathcal{D}$, then*

$$\mathcal{D} \subseteq \mathcal{C} \iff B \in \mathcal{C}.$$

   In particular, if B is NP-complete, then

$$\mathrm{P} = \mathrm{NP} \iff B \in \mathrm{P}.$$

6. *For each nontrivial set $B \in \mathrm{P}$ (i.e., $\emptyset \neq B \neq \Sigma^*$) and for each set $A \in \mathrm{P}$, $A \leq_{\mathrm{m}}^{\mathrm{p}} B$. Thus, every nontrivial set in P is $\leq_{\mathrm{m}}^{\mathrm{p}}$-complete for P.*

Reminder: Properties of $\leq_m^p$

Proof: All these properties follow easily from the definitions.

As examples, we only show two selected properties:

3. We show that P is $\leq_m^p$-closed:
   - Let $A \leq_m^p B$ via $f \in \mathrm{FP}$, where f is computed by DPTM M running in time $p \in \mathbb{P}\mathrm{ol}$, and
   - let $B \in \mathrm{P}$ via DPTM N running in time $q \in \mathbb{P}\mathrm{ol}$.

   To show that $A \in \mathrm{P}$, given input x, simply
   - compute $f(x)$ via M,
   - run N on input $f(x)$, and
   - accept if and only if $N(f(x))$ accepts.

   Note that $|f(x)|$ is polynomial in $|x|$, and as $p, q \in \mathbb{P}\mathrm{ol}$, so is $p(q)$.

Reminder: Properties of $\leq_m^p$

6. To show that every nontrivial set B in P is $\leq_m^p$-complete for P, choose
   - a string $b \in B$ and
   - a string $\bar{b} \notin B$

   (which is possible because $\emptyset \neq B \neq \Sigma^*$).

   Let A be an arbitrary set in P.

   Define the reduction

$$f(x) = \begin{cases} b & \text{if } x \in A \\ \bar{b} & \text{if } x \notin A. \end{cases}$$

   Clearly, $f \in \mathrm{FP}$ and f witnesses that $A \leq_m^p B$. ❑

Reminder: Log-Space Many-One Reducibility

Definition

- Let $\Sigma = \{0, 1\}$ be a fixed alphabet, and let $A, B \subseteq \Sigma^*$.
- Let FL denote the set of log-space computable total functions mapping from Σ^* to Σ^*.
- Let $\mathcal{C}$ be any complexity class.

1. Define the *log-space many-one reducibility*, denoted by $\leq_{\mathrm{m}}^{\log}$, as follows: $A \leq_{\mathrm{m}}^{\log} B$ if there is a function $f \in \mathrm{FL}$ such that

$$(\forall x \in \Sigma^*)\, [x \in A \iff f(x) \in B].$$

Reminder: Log-Space Many-One Reducibility

Definition (continued)

2. A set B is *$\leq_{\mathrm{m}}^{\log}$-hard for $\mathcal{C}$* if $A \leq_{\mathrm{m}}^{\log} B$ for each $A \in \mathcal{C}$.

3. A set B is *$\leq_{\mathrm{m}}^{\log}$-complete for $\mathcal{C}$* if
   1. B is $\leq_{\mathrm{m}}^{\log}$-hard for $\mathcal{C}$ (lower bound) and
   2. $B \in \mathcal{C}$ (upper bound).

4. $\mathcal{C}$ is said to be *closed under the $\leq_{\mathrm{m}}^{\log}$-reducibility* ($\leq_{\mathrm{m}}^{\log}$-*closed*, for short) if for any two sets A and B,

$$\text{if } A \leq_{\mathrm{m}}^{\log} B \text{ and } B \in \mathcal{C}, \text{ then } A \in \mathcal{C}.$$

Reminder: Properties of $\leq_{\mathrm{m}}^{\log}$

Theorem

The $\leq_{\mathrm{m}}^{\log}$-reducibility is a transitive relation.

Reminder: The Satisfiability Problem

- A boolean formula φ is in *conjunctive normal form* (*CNF*, for short) if and only if φ is of the form

$$\begin{aligned}\varphi(x_1, x_2, \ldots, x_n) &= \bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{k_i} \ell_{i,j} \right) \\ &= (\ell_{1,1} \vee \cdots \vee \ell_{1,k_1}) \wedge \cdots \wedge (\ell_{m,1} \vee \cdots \vee \ell_{m,k_m}),\end{aligned}$$

  where the $\ell_{i,j}$ are literals over $\{x_1, x_2, \ldots, x_n\}$, and the disjuncts $\left(\bigvee_{j=1}^{k_i} \ell_{i,j}\right)$ of literals are said to be the *clauses of φ*.

- A boolean formula φ is in *k-CNF* if and only if φ is in CNF and each clause of φ has at most k literals.

- Analogously: *disjunctive normal form* (*DNF*, for short) and *k-DNF*.

Reminder: The Satisfiability Problem

Definition

Define the decision problems

$$\begin{aligned} \text{SAT} &= \left\{ \varphi \,\middle|\, \varphi \text{ is a satisfiable boolean formula} \right\}, \\ k\text{-SAT} &= \left\{ \varphi \,\middle|\, \varphi \text{ is a satisfiable boolean formula in } k\text{-CNF} \right\}. \end{aligned}$$

Remark:

- 2-SAT is $\leq_{\mathrm{m}}^{\mathrm{log}}$-complete for NL, the class of problems solvable in nondeterministic logarithmic space.
  As $\mathrm{NL} \subseteq \mathrm{P}$, it follows that 2-SAT is in P.
- SAT is easy to solve (i.e., in P) for formulas in DNF.

Reminder: SAT is NP-complete

Theorem (Cook 1971 & Levin 1973)

SAT *is* $\leq_{m}^{p}$*-complete for* NP.

Proof:

1. SAT $\in \mathrm{NP}$: Given a boolean formula φ with variable set X,
   1. guess nondeterministically a truth assignment
   $$T : X \to \{\textbf{true}, \textbf{false}\},$$
   2. check deterministically whether $T \models \varphi$ and accept accordingly.

Cook Reduction: Boolean Variables in F_x

2. SAT is NP-hard: To show $A \leq_{\mathrm{m}}^{\mathrm{p}} \mathrm{SAT}$ for any NP set A (with $L(M) = A$ for NPTM M), construct a boolean formula F_x such that:

$$x \in A \iff f(x) = F_x \in \mathrm{SAT}. \quad (1)$$

Let $x = x_1 x_2 \cdots x_n$ be the input string, where $x_i \in \Sigma$ for each i.

Since $M = (\Sigma, \Gamma, Z, \delta, \square, s_0, F)$ works in, w.l.o.g., time *exactly* $p(n)$, the tape head can move no further than $p(n)$ tape cells to the left or right.

Enumerate the relevant tape cells from $-p(n)$ through $p(n)$.

Start configuration of $M(x)$:

- input symbols $x_1 x_2 \cdots x_n$ in tape cells 0 through $n - 1$,
- the head currently scans the tape cell with number 0, and
- the start state is s_0.

Cook Reduction: Boolean Variables in F_x

$\cdots$	$\square$	$\cdots$	$\square$	x_1	x_2	$\cdots$	x_n	$\square$	$\cdots$	$\square$	$\cdots$
$\cdots$	$-p(n)$	$\cdots$	-1	0	1	$\cdots$	$n-1$	n	$\cdots$	$p(n)$	$\cdots$

variables of F_x	index range	intended meaning
$\mathrm{state}_{t,s}$	$0 \leq t \leq p(n)$ $s \in Z$	true $\iff$ in step t, M is in state s
$\mathrm{head}_{t,i}$	$0 \leq t \leq p(n)$ $-p(n) \leq i \leq p(n)$	true $\iff$ in step t, M's head scans cell i
$\mathrm{tape}_{t,i,a}$	$0 \leq t \leq p(n)$ $-p(n) \leq i \leq p(n)$ $a \in \Gamma$	true $\iff$ in step t, the symbol a is in cell i of M's tape

Cook Reduction: Structure of F_x

$F_x = S \wedge T_1 \wedge T_2 \wedge E \wedge C$, where

- S: correct *start* of the computation of $M(x)$;
- T_1: correct *transition* from step t to step $t+1$ for those tape cells whose contents can be altered by the head of M;
- T_2: correct *transition* from step t to step $t+1$ for those tape cells whose contents cannot be altered by the head of M;

Cook Reduction: Structure of F_x

$F_x = S \wedge T_1 \wedge T_2 \wedge E \wedge C$, where

- *E*: correct *end* of the computation of $M(x)$, i.e., E is true if and only if $M(x)$ has an accepting computation path;
- *C*: general *correctness*, i.e.,
  C is true if and only if the following conditions hold:
  - in each step t of $M(x)$, there exists *exactly* one state $s \in Z$ such that $\mathrm{state}_{t,s}$ is true, and there exists *exactly* one i such that $\mathrm{head}_{t,i}$ is true;
  - in each step t of $M(x)$ and for each cell number i, there exists *exactly* one $a \in \Gamma$ such that $\mathrm{tape}_{t,i,a}$ is true.

Cook Reduction: Subformula C

Let the set of states and the working alphabet of M be given by

$$\begin{aligned} Z &= \{s_0, s_1, \ldots, s_k\}, \\ \Gamma &= \{\Box, a_1, a_2, \ldots, a_\ell\}. \end{aligned}$$

Define

$$\begin{aligned} C = \bigwedge_{0 \leq t \leq p(n)} & [D(\mathrm{state}_{t,s_0}, \mathrm{state}_{t,s_1}, \ldots, \mathrm{state}_{t,s_k}) \wedge \\ & D(\mathrm{head}_{t,-p(n)}, \mathrm{head}_{t,-p(n)+1}, \ldots, \mathrm{head}_{t,p(n)}) \wedge \\ & \bigwedge_{-p(n) \leq i \leq p(n)} D(\mathrm{tape}_{t,i,\Box}, \mathrm{tape}_{t,i,a_1}, \ldots, \mathrm{tape}_{t,i,a_\ell})], \end{aligned}$$

where the structure of the three subformulas D of C above is specified in the next lemma.

Cook Reduction: Lemma for Subformula C

Lemma

For each $m \geq 1$, there exists a boolean formula D in the variables $v_1, v_2, \ldots, v_m$ such that:

- *$D(v_1, v_2, \ldots, v_m)$ is true if and only if* exactly *one variable v_i is true, and*
- *the size of the formula D (i.e., the number of variable occurrences in D) is in $\mathcal{O}(m^2)$.*

Proof of Lemma. For fixed $m \geq 1$, define

$$D(v_1, v_2, \ldots, v_m) = \underbrace{\left(\bigvee_{i=1}^{m} v_i \right)}_{D_{\geq}} \wedge \underbrace{\left(\bigwedge_{j=1}^{m-1} \bigwedge_{k=j+1}^{m} \neg(v_j \wedge v_k) \right)}_{D_{\leq}}.$$

Cook Reduction: Lemma for Subformula C

Subformulas $D_{\geq}$ and $D_{\leq}$ of D satisfy the following properties:

$$D_{\geq}(v_1, v_2, \ldots, v_m) \text{ is true} \iff \text{at least one variable } v_i \text{ is true;} \quad (2)$$

$$D_{\leq}(v_1, v_2, \ldots, v_m) \text{ is true} \iff \text{at most one variable } v_i \text{ is true.} \quad (3)$$

Equation (2) is obvious. To see that also (3) is true, observe that the formula $D_{\leq}(v_1, v_2, \ldots, v_m)$ has the following structure:

$$\begin{array}{ccccccccc}
(\neg v_1 \vee \neg v_2) & \wedge & (\neg v_1 \vee \neg v_3) & \wedge & \cdots & \wedge & (\neg v_1 \vee \neg v_m) \\
 & \wedge & (\neg v_2 \vee \neg v_3) & \wedge & \cdots & \wedge & (\neg v_2 \vee \neg v_m) \\
 & & & & \ddots & & \vdots \\
 & & & & & \wedge & (\neg v_{m-1} \vee \neg v_m).
\end{array}$$

(2) and (3) together imply that $D(v_1, v_2, \ldots, v_m)$ is true if and only if exactly one v_i is true. Clearly, the size of D is in $\mathcal{O}(m^2)$. ❑ Lemma

Cook Reduction: Subformulas T_1 and T_2

Letting δ denote M's transition function and $y \in \{-1, 0, 1\}$ represent M moving its head to the left, to the right, or not at all, respectively, define

$$T_1 = \bigwedge_{t,s,i,a} \Big(\big(\mathrm{state}_{t,s} \wedge \mathrm{head}_{t,i} \wedge \mathrm{tape}_{t,i,a}\big) \Longrightarrow \bigvee_{\substack{\hat{s} \in Z,\, \hat{a} \in \Gamma,\, y \in \{-1,0,1\} \\ \text{with } (\hat{s}, \hat{a}, y) \in \delta(s,a)}} \big(\mathrm{state}_{t+1,\hat{s}} \wedge \mathrm{head}_{t+1,i+y} \wedge \mathrm{tape}_{t+1,i,\hat{a}}\big)\Big)$$

and

$$T_2 = \bigwedge_{t,i,a} \Big(\big(\neg \mathrm{head}_{t,i} \wedge \mathrm{tape}_{t,i,a}\big) \Longrightarrow \mathrm{tape}_{t+1,i,a}\Big) .$$

Cook Reduction: Subformulas S and E

Define

$$S = \mathrm{state}_{0,s_0} \wedge \mathrm{head}_{0,0} \wedge \bigwedge_{i=-p(n)}^{-1} \mathrm{tape}_{0,i,\Box} \wedge \bigwedge_{i=0}^{n-1} \mathrm{tape}_{0,i,x_{i+1}} \wedge \bigwedge_{i=n}^{p(n)} \mathrm{tape}_{0,i,\Box}$$

and

$$E = \bigvee_{s \in F} \mathrm{state}_{p(n),s} .$$

Cook Reduction: Proof of Equivalence (1)

We show:

$$x \in A \iff f(x) = F_x \in \mathrm{SAT}.$$

(⇒)

$$
\begin{aligned}
x \in A \quad &\Rightarrow \quad \text{there exists an accepting computation path } \alpha \text{ of } M(x) \\
&\Rightarrow \quad \text{assigning truth values to every variable of } F_x \\
& \quad \text{according to } \alpha\text{, associating with each variable} \\
& \quad \text{its "intended meaning" according to our table, this} \\
& \quad \text{truth assignment satisfies each subformulas of } F_x \\
&\Rightarrow \quad F_x \in \mathrm{SAT}
\end{aligned}
$$

Cook Reduction: Proof of Equivalence (1)

We show:

$$x \in A \iff f(x) = F_x \in \mathrm{SAT}.$$

(⇐)

$F_x \in \mathrm{SAT}$ $\Rightarrow$ there exists a truth assigment τ to F_x's variables satisfying F_x

$\Rightarrow$ according to τ, the variables $\mathrm{state}_{t,s}$, $\mathrm{head}_{t,i}$, and $\mathrm{tape}_{t,i,a}$ of F_x can be sensibly interpreted as a sequence of configurations $K_0, K_1, \ldots, K_{p(n)}$ of $M(x)$ along some accepting computation path of $M(x)$

$\Rightarrow$ $x \in A$

Cook Reduction: Reduction is in FP

Finally, to show $f \in \mathrm{FP}$, note that:

- The size of F_x is polynomial in $n = |x|$:

$$|F_x| \in \mathcal{O}((p(n))^3).$$

- An FP algorithm computing f runs in time linear in $|F_x|$. ❑

Reminder: 3-SAT is NP-complete

Theorem

3-SAT *is* $\leq_{\mathrm{m}}^{\mathrm{p}}$*-complete for* NP*.*

Proof: Membership in NP for the restricted problem follows immediately from that for the general problem.

To prove that SAT $\leq_{\mathrm{m}}^{\mathrm{p}}$ 3-SAT, define a reduction f mapping any given boolean formula φ to a boolean formula ψ in 3-CNF such that:

$$\varphi \text{ is satisfiable} \iff \psi \text{ is satisfiable.} \quad (4)$$

Let

$$\varphi(x_1, x_2, \ldots, x_n) = C_1 \wedge C_2 \wedge \cdots \wedge C_m,$$

where the C_j are the clauses of φ.

Reminder: 3-SAT is NP-complete

The formula ψ is constructed from φ as follows.

The variables of ψ are φ's variables $x_1, x_2, \ldots, x_n$ and, for each clause C_j of φ, the variables $y_1^j, y_2^j, \ldots, y_{h_j}^j$.

Define

$$\psi = D_1 \wedge D_2 \wedge \cdots \wedge D_m,$$

where each subformula D_j of ψ is constructed from the clause C_j of φ as follows.

Consider the j^{th} clause of φ, and suppose that $C_j = (z_1 \vee z_2 \vee \cdots \vee z_k)$, where each z_i is a literal over $\{x_1, x_2, \ldots, x_n\}$.

Distinguish the following four cases.

Reminder: 3-SAT is NP-complete

Case 1: $k = 1$. Define

$$D_j = (z_1 \vee y_1^j \vee y_2^j) \wedge (z_1 \vee \neg y_1^j \vee y_2^j) \wedge (z_1 \vee y_1^j \vee \neg y_2^j) \wedge (z_1 \vee \neg y_1^j \vee \neg y_2^j).$$

Case 2: $k = 2$. Define

$$D_j = (z_1 \vee z_2 \vee y_1^j) \wedge (z_1 \vee z_2 \vee \neg y_1^j).$$

Case 3: $k = 3$. Define $D_j = C_j = (z_1 \vee z_2 \vee z_3)$.

Case 4: $k \geq 4$. Define

$$\begin{aligned} D_j \;=\; & (z_1 \vee z_2 \vee y_1^j) \wedge (\neg y_1^j \vee z_3 \vee y_2^j) \wedge (\neg y_2^j \vee z_4 \vee y_3^j) \wedge \cdots \wedge \\ & (\neg y_{k-4}^j \vee z_{k-2} \vee y_{k-3}^j) \wedge (\neg y_{k-3}^j \vee z_{k-1} \vee z_k). \end{aligned}$$

Reminder: 3-SAT is NP-complete

Observe that the reduction f is polynomial-time computable.

It remains to show that (4) is true.

(⇒) Let t be a truth assignment to the variables $x_1, x_2, \ldots, x_n$ of φ such that $t(\varphi) = 1$.

Extend t to a truth assignment t' of the variables of ψ as follows.

Since for $i \neq j$, the subformulas D_i and D_j are disjoint with respect to the y variables, it is enough to consider all subformulas of ψ separately. Consider the subformula D_j for any fixed j.

In Cases 1 through 3 above, t already satisfies D_j, so t can arbitrarily be extended to t'.

Reminder: 3-SAT is NP-complete

Consider Case 4 above.

Let z_i, where $1 \leq i \leq k$ be the first literal in C_j for which $t(z_i) = 1$.

Such an i must exist, since t satisfies C_j.

If $i \in \{1, 2\}$, then set $t'(y_\ell^j) = 0$ for each ℓ with $1 \leq \ell \leq k - 3$.

If $i \in \{k-1, k\}$, then set $t'(y_\ell^j) = 1$ for each ℓ with $1 \leq \ell \leq k - 3$.

Otherwise, set

$$t'(y_\ell^j) = \begin{cases} 1 & \text{if } 1 \leq \ell \leq i-2 \\ 0 & \text{if } i-1 \leq \ell \leq k-3. \end{cases}$$

In each case, t' satisfies D_j.

Hence, $t'(\psi) = 1$, so ψ is satisfiable.

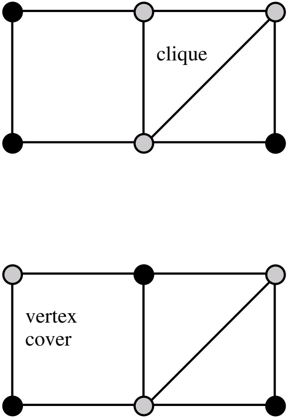
Reminder: 3-SAT is NP-complete

($\Leftarrow$) Let t' be a satisfying truth assignment to ψ.

Let t be the restriction of t' to the variables $x_1, x_2, \ldots, x_n$ of φ.

Hence, $t(\varphi) = 1$, so φ is satisfiable, which proves (4) and the theorem. ❑

Reminder: Clique, Independent Set, Vertex Cover, and Dominating Set



Reminder: Clique and Independent Set

Definition

Let G be an undirected graph.

- A *clique of G* is a subset $C \subseteq V(G)$ such that for any two vertices $x, y \in C$ with $x \neq y$,
$$\{x, y\} \in E(G).$$

- An *independent set of G* is a subset $I \subseteq V(G)$ such that for any two vertices $x, y \in I$ with $x \neq y$,
$$\{x, y\} \notin E(G).$$

Reminder: Vertex Cover and Dominating Set

Definition

Let G be an undirected graph.

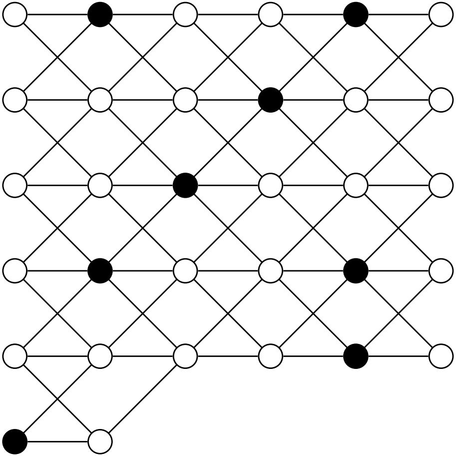
- A *vertex cover of G* is a subset $C \subseteq V(G)$ such that for each edge $\{x, y\} \in E(G)$,

$$\{x, y\} \cap C \neq \emptyset.$$

- A *dominating set of G* is a subset $D \subseteq V(G)$ such that for each $x \in V(G) - D$ there exists a vertex $y \in D$ such that

$$\{x, y\} \in E.$$

Reminder: Dominating Set



Reminder: Clique, Independent Set, Vertex Cover, and Dominating Set

Definition

$$\begin{aligned}
\textsc{Clique} &= \{(G,k) \mid G \text{ has a clique of size } \geq k\} \\
\textsc{Independent Set} &= \{(G,k) \mid G \text{ has an independent set of size } \geq k\} \\
\textsc{Vertex Cover} &= \{(G,k) \mid G \text{ has a vertex cover of size } \leq k\} \\
\textsc{Dominating Set} &= \{(G,k) \mid G \text{ has a dominating set of size } \leq k\}
\end{aligned}$$

Reminder: Clique, Independent Set, and Vertex Cover

Lemma

For each graph G and for each subset $U \subseteq V(G)$, the following are equivalent:

1. *U is a vertex cover of G.*
2. *$\overline{U} = V(G) - U$ is an independent set of G.*
3. *$\overline{U} = V(G) - U$ is a clique of the co-graph of G, which is defined as the graph with vertex set $V(G)$ and edge set*

$$\{\{u, v\} \mid u, v \in V(G) \text{ and } \{u, v\} \notin E(G)\}.$$

Reminder: Clique, Independent Set, and Vertex Cover

Theorem

CLIQUE, INDEPENDENT SET, *and* VERTEX COVER *are* NP*-complete.*

Proof: It is easy to see that each of CLIQUE, INDEPENDENT SET, and VERTEX COVER belongs to NP.

The previous lemma implies that these three problems are pair-wise $\leq_{\mathrm{m}}^{\mathrm{p}}$-equivalent:

$$\text{CLIQUE} \leq_{\mathrm{m}}^{\mathrm{p}} \text{INDEPENDENT SET} \leq_{\mathrm{m}}^{\mathrm{p}} \text{VERTEX COVER} \leq_{\mathrm{m}}^{\mathrm{p}} \text{CLIQUE}.$$

Hence, it suffices to prove that, e.g., 3-SAT $\leq_{\mathrm{m}}^{\mathrm{p}}$ INDEPENDENT SET.

Proof Idea: Independent Set is NP-complete

Let $\varphi(x_1, x_2, \ldots, x_n) = C_1 \wedge C_2 \wedge \cdots \wedge C_m$ be a given boolean formula with exactly three literals per clause.

For each i with $1 \leq i \leq m$, let the i^{th} clause be given by $C_i = (z_{i,1} \vee z_{i,2} \vee z_{i,3})$, where every $z_{i,j} \in \{x_1, x_2, \ldots, x_n\} \cup \{\neg x_1, \neg x_2, \ldots, \neg x_n\}$ is a literal.

The reduction f maps φ to the pair (G, m), where G is the graph with vertex set

$$V(G) = \{z_{i,j} \mid 1 \leq i \leq m \text{ and } 1 \leq j \leq 3\}$$

and edge set

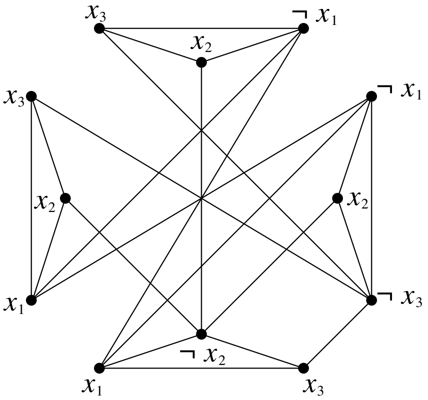
$$E(G) = \{\{z_{i,j}, z_{i,k}\} \mid 1 \leq i \leq m \text{ and } 1 \leq j,k \leq 3 \text{ and } j \neq k\} \cup \{\{z_{i,j}, z_{r,s}\} \mid i \neq r \text{ and } z_{i,j} = \neg z_{r,s}\}.$$

Proof Idea: Independent Set is NP-complete

According to the construction, formula

$$\varphi(x_1, x_2, x_3) = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (x_1 \vee \neg x_2 \vee x_3)$$

is transformed into the following graph:



Proof Idea: Independent Set is NP-complete

Clearly, $f \in \mathrm{FP}$. The construction implies that:

$\varphi \in$ 3-SAT $\iff$ there exists a truth assignment t with $t(\varphi) = 1$

$\iff$ every clause C_i has a literal z_{i,j_i} with $t(z_{i,j_i}) = 1$

$\iff$ there exists a sequence of literals $z_{1,j_1}, \ldots, z_{m,j_m}$ such that $z_{i,j_i} \neq \neg z_{k,j_k}$ for $i, k \in \{1, \ldots, m\}$ with $i \neq k$

$\iff$ there exists a sequence of literals $z_{1,j_1}, \ldots, z_{m,j_m}$ such that $\{z_{1,j_1}, \ldots, z_{m,j_m}\}$ is an independent set of size m in G.

Since G has an independent set of size at least m if and only if φ is satisfiable, the reduction f witnesses that

$$\text{3-SAT} \leq_{\mathrm{m}}^{\mathrm{p}} \textsc{Independent Set}.$$

❑

Reminder: Dominating Set is NP-complete

Theorem

DOMINATING SET *is* NP*-complete.*

Proof: Exercise. Hint: Reduction from VERTEX COVER. ❑

Reminder: Graph Colorability

Definition

Let $G = (V(G), E(G))$ be an undirected graph.

- A *k-coloring of G* is a mapping $V(G) \rightarrow \{1, 2, \ldots, k\}$.
- A k-coloring ψ of G is called *legal* if for any two vertices x and y in $V(G)$, if $\{x, y\} \in E(G)$ then $\psi(x) \neq \psi(y)$.
- The *chromatic number of G*, denoted by $\chi(G)$, is the smallest number k such that G is legally k-colorable.

Reminder: Graph Colorability

Definition

For fixed $k \geq 1$, define

$$k\text{-}\textsc{Color} = \{G \mid G \text{ is a graph with } \chi(G) \leq k\}.$$

Example:

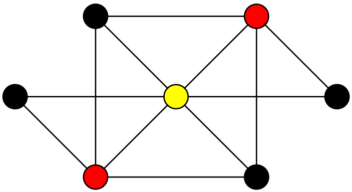


Figure: A 3-colorable graph

Reminder: 3-COLOR is NP-complete

Fact

2-COLOR is in P. **without proof**

Theorem

3-COLOR is NP-complete.

Proof:

1. 3-COLOR $\in$ NP is easy to see.
2. 3-COLOR is NP-hard: We show 3-SAT $\leq_{m}^{p}$ 3-COLOR. Let

$$\varphi(x_1, x_2, \ldots, x_n) = C_1 \wedge C_2 \wedge \cdots \wedge C_m$$

be a given 3-SAT instance with exactly three literals per clause.

Reminder: 3-COLOR is NP-complete

Define a reduction f mapping φ to the graph G constructed as follows. The vertex set of G is defined by

$$\begin{aligned} V(G) &= \{v_1, v_2, v_3\} \cup \{x_i, \neg x_i \mid 1 \leq i \leq n\} \\ &\cup \{y_{j,k} \mid 1 \leq j \leq m \text{ and } 1 \leq k \leq 6\}, \end{aligned}$$

where the x_i and $\neg x_i$ are vertices representing the literals x_i and their negations $\neg x_i$, respectively.

Reminder: 3-COLOR is NP-complete

The edge set of G is defined by

$$
\begin{aligned}
E(G) \;=\;& \{\{v_1, v_2\}, \{v_2, v_3\}, \{v_1, v_3\}\} \cup \{\{x_i, \neg x_i\} \mid 1 \leq i \leq n\} \\
& \cup \{\{v_3, x_i\}, \{v_3, \neg x_i\} \mid 1 \leq i \leq n\} \\
& \cup \{\{a_j, y_{j,1}\}, \{b_j, y_{j,2}\}, \{c_j, y_{j,3}\} \mid 1 \leq j \leq m\} \\
& \cup \{\{v_2, y_{j,6}\}, \{v_3, y_{j,6}\} \mid 1 \leq j \leq m\} \\
& \cup \{\{y_{j,1}, y_{j,2}\}, \{y_{j,1}, y_{j,4}\}, \{y_{j,2}, y_{j,4}\} \mid 1 \leq j \leq m\} \\
& \cup \{\{y_{j,3}, y_{j,5}\}, \{y_{j,3}, y_{j,6}\}, \{y_{j,5}, y_{j,6}\} \mid 1 \leq j \leq m\} \\
& \cup \{\{y_{j,4}, y_{j,5}\} \mid 1 \leq j \leq m\},
\end{aligned}
$$

where $a_j, b_j, c_j \in \bigcup_{1 \leq i \leq n} \{x_i, \neg x_i\}$ are vertices representing the literals occurring in clause $C_j = (a_j \vee b_j \vee c_j)$.

Skeletal Structure of the Graph in 3-SAT $\leq_{\rm m}^{\rm p}$ 3-COLOR

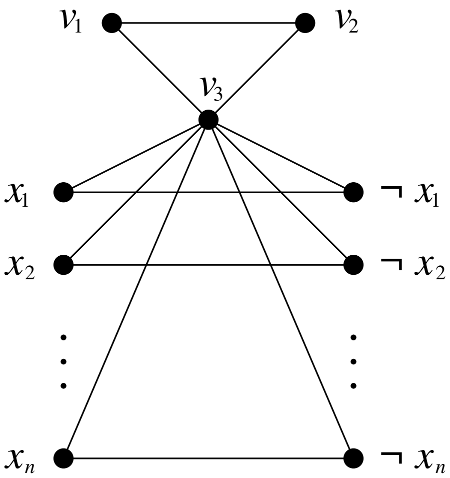


Figure: Skeletal Structure of the graph in 3-SAT $\leq_{\rm m}^{\rm p}$ 3-COLOR

Clause Graph in 3-SAT $\leq_m^p$ 3-COLOR

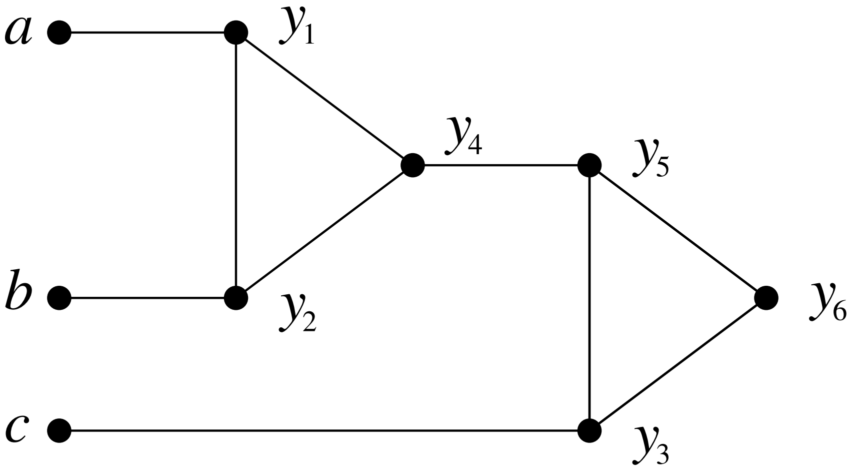


Figure: Graph H for clause $C = (a \vee b \vee c)$ in 3-SAT $\leq_m^p$ 3-COLOR

Properties of Clause Graph H in 3-SAT $\leq_{\mathrm{m}}^{\mathrm{p}}$ 3-COLOR

1. Vertices x_i and $\neg x_i$ corresponding to the literals x_i and $\neg x_i$ are legally colored 1 (“true”) or 2 (“false”).

2. Any coloring of the vertices a, b, and c that assigns color 1 to one of a, b, and c can be extended to a legal 3-coloring of H that assigns color 1 to y_6. Thus, if $\varphi \in$ 3-SAT then $G \in$ 3-COLOR.

3. If ψ is a legal 3-coloring of H with $\psi(a) = \psi(b) = \psi(c) = i$, then $\psi(y_6) = i$. Thus, if $\varphi \notin$ 3-SAT then $G \notin$ 3-COLOR.

It follows that

$$\varphi \in \text{3-SAT} \iff f(\varphi) = G \in \text{3-COLOR}$$

Clearly, reduction f is polynomial-time computable. ❑

Directed Hamilton Circuit

Definition

DIRECTED HAMILTON CIRCUIT (DHC, for short) is the following problem:

Given: A directed graph $G = (V(G), E(G))$.

Question: Does there exist a *Hamilton cycle* in G, i.e., a sequence $(v_1, v_2, \ldots, v_n)$, $v_i \in V(G)$, $n = |V(G)|$, such that $(v_n, v_1) \in E(G)$ and $(v_i, v_{i+1}) \in E(G)$ for $1 \leq i < n$?

Theorem

DHC *is* NP*-complete.* **without proof**

Hamilton Circuit

Definition

HAMILTON CIRCUIT (HC, for short) is the following problem:

Given: An undirected graph $G = (V(G), E(G))$.

Question: Does there exist a *Hamilton cycle* in G, i.e., a sequence $(v_1, v_2, \ldots, v_n)$, $v_i \in V(G)$, $n = |V(G)|$, such that $\{v_n, v_1\} \in E(G)$ and $\{v_i, v_{i+1}\} \in E(G)$ for $1 \leq i < n$?

Theorem

HC *is* NP*-complete.*

Proof: Excercise. Hint: Reduction from DHC. ❑

Traveling Salesperson Problem

Definition

The TRAVELING SALESPERSON PROBLEM (TSP, for short) is the following problem:

Given: A complete undirected graph $K_n = (V, E)$, a cost function $c : E \to \mathbb{N}$, and $k \in \mathbb{N}$.

Question: Does there exist a Hamilton cycle in K_n such that the sum of the edge costs is at most k?

Theorem

TSP *is* NP*-complete.*

Proof: TSP $\in \mathrm{NP}$ is easy to see.

Traveling Salesperson Problem is NP-complete

TSP is NP-hard: We show HC $\leq_{\mathrm{m}}^{\mathrm{p}}$ TSP.

Given an undirected graph $G = (V(G), E(G))$ with $V(G) = \{v_1, v_2, \ldots, v_n\}$, define

$$f(G) = (K_n, c, n),$$

where $K_n = (V, E)$, $V = \{1, 2, \ldots, n\}$, and for each edge $e = \{i, j\}$ of K_n:

$$c(\{i, j\}) = \begin{cases} 1 & \text{if } \{v_i, v_j\} \in E(G) \\ 2 & \text{otherwise.} \end{cases}$$

Clearly, $G \in$ HC if and only if $f(G) \in$ TSP. ❑