Chapter 2 - Reductions and Complete Problems

- polynomial time reductions
- complete problems for classes **NP** and **PSPACE**

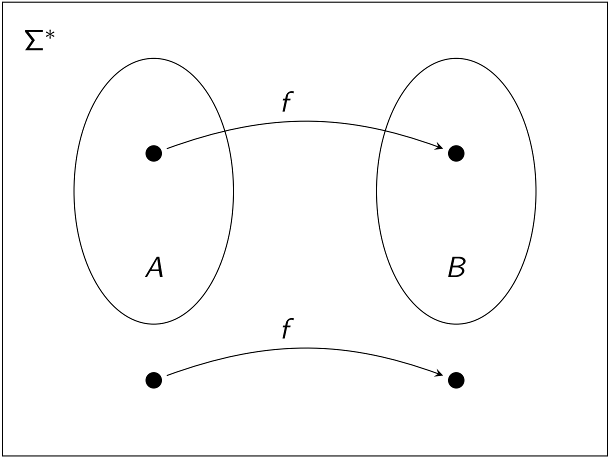
Polynomial time computable functions and reductions

Definition 2.1
A function $f : \Sigma^* \rightarrow \Sigma^*$ *is a polynomial time computable function if some polynomial time deterministic Turing machine* M *exists that halts with* $\triangleright f(w)$ *on its tape, when started on any input* $w \in \Sigma^*$*.*

Definition 2.2
Language A *is polynomial time mapping reducible, or simply polynomial time reducible, to language* B*, written* $A \leq_P B$*, if a polynomial time computable function* $f : \Sigma^* \rightarrow \Sigma^*$ *exists, where for every* $w \in \Sigma^*$

$$w \in A \Leftrightarrow f(w) \in B.$$

Illustration of polynomial time reductions



Properties of polynomial reductions

Theorem 2.3

If $A \leq_P B$ and $B \in \mathbf{P}$, then $A \in \mathbf{P}$.

From B to A

M polynomial time DTM deciding B.

$N =$ "On input w:
1. Compute $f(w)$.
2. Run M on input $f(w)$, and output whatever M outputs."

Lemma 2.4

If $A \leq_P B$ and $B \leq_P C$, then $A \leq_P C$.

CNF-formulas

Formulas in conjunctive normal form and cliques

- a literal is a Boolean variable x or a negated Boolean variable $\neg x$ or $\bar{x}$
- a clause consists of several literals connected with $\vee$'s, e.g. $(x_1 \vee \bar{x}_2 \vee x_4)$.
- a Boolean formula is in conjunctive normal form, called a *cnf-formula* if it comprises several clauses connected with $\wedge$'s, e.g. $(x_1 \vee \bar{x}_2 \vee x_4) \wedge (x_2 \vee \bar{x}_5 \vee x_6) \wedge (x_3 \vee \bar{x}_6)$.
- a cnf-formula is a *3cnf-formula* if all its clauses have three literals, e.g. $(x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (x_1 \vee \bar{x}_2 \vee x_4) \wedge (x_2 \vee \bar{x}_5 \vee x_6)$.
- a *clique* in an undirected graph $G = (V, E)$ is a subset $C \subseteq V$ of vertices such that for any two vertices $u, v \in C$ $(u, v) \in E$
- a clique C is a *k-clique*, if $|C| = k$

The languages *3SAT* and *CLIQUE*

3SAT

$$\mathit{3SAT} = \{\langle \phi \rangle \mid \phi \text{ is a satisfiable 3cnf-formula}\}$$

CLIQUE

$$\mathit{CLIQUE} = \{\langle G, k \rangle \mid G \text{ is an undirected graph with a } k\text{-clique}\}$$

Theorem 2.5

3SAT is polynomial time reducible to CLIQUE.

Reduction from *3SAT* to *CLIQUE*

Input

A 3cnf-formula with k clauses

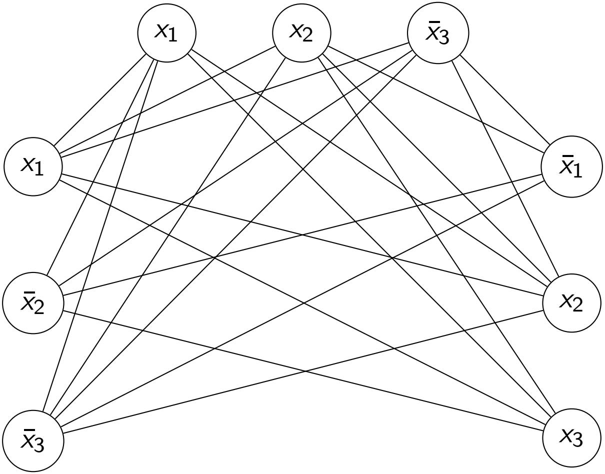
$$\phi = (a_1 \vee b_1 \vee c_1) \wedge (a_2 \vee b_2 \vee c_2) \wedge \cdots \wedge (a_k \vee b_k \vee c_k).$$

Reduction

- $G = (V, E)$ contains $3k$ vertices organized in k triples $t_1, \ldots, t_k$, one for each clause in ϕ. Vertices in a triple correspond to literals in the clause and are labeled with the corresponding literal.
- Any two vertices are connected by an edge in G, except if
  1. they belong to the same triple, or
  2. their labels are negations of each other.
- Size of clique set to k.

Example for the reduction from *3SAT* to *CLIQUE*

Graph to formula $(x_1 \vee x_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee x_2 \vee x_3)$:



Complete problems

Definition 2.6

A language B is **NP***-complete if it satisfies two conditions:*

1. *B is in* **NP***, and*
2. *every language A in* **NP** *is polynomial time reducible to B.*

Definition 2.7

A language B is **PSPACE***-complete if it satisfies two conditions:*

1. *B is in* **PSPACE***, and*
2. *every language A in* **PSPACE** *is polynomial time reducible to B.*

Fundamental properties of complete langages

Theorem 2.8

1. *If B is* **NP***-complete and $B \in$* **P***, then* **P** $=$ **NP***.*
2. *If B is* **PSPACE***-complete and $B \in$* **P***, then* **P** $=$ **PSPACE***.*

Theorem 2.9

1. *If B is* **NP***-complete and $B \leq_P C$ for C in* **NP***, then C is* **NP***-complete.*
2. *If B is* **PSPACE***-complete and $B \leq_P C$ for C in* **PSPACE***, then C is* **PSPACE***-complete.*

The basic complete languages - *SAT* and *TQBF*

The languages

- $SAT = \{\langle \phi \rangle \mid \phi \text{ is a satisfiable Boolean formula}\}$
- $TQBF = \{\langle \phi \rangle \mid \phi \text{ is a true fully quantified Boolean formula}\}$

Theorem 2.10 (Cook-Levin)

SAT is **NP***-complete.*

Theorem 2.11

TQBF is **PSPACE***-complete.*

Proofs for Theorems 2.10 and 2.11

Proof idea

- $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ polynomial time NTM or polynomial space DTM, $w \in \Sigma^*$
- Construct Boolean formula ϕ or fully quantified Boolean formula ϕ that simulates computation of M on input w.
- If M is a NTM, then $w \in L(M)$ iff ϕ has a satisfying assignment.
- If M is a polynomial space DTM, then $w \in L(M)$ iff ϕ is true.
- Difference between proofs for two theorems only at the end.

Proof preliminaries

- Let M be a $t(n) - 1$ time and $s(n)$ space TM and set $A := Q \cup \Gamma$.
- Every configuration c of M on input w can be identified with an element of $A^{s(n)+1}$, where $n = |w|$.
- Use four predicates on elements in $A^{s(n)+1}$:

$$
\begin{aligned}
\text{legal} : A^{s(n)+1} &\to \{0,1\} \\
\text{start} : A^{s(n)+1} &\to \{0,1\} \\
\text{accept} : A^{s(n)+1} &\to \{0,1\} \\
\text{succ} : A^{s(n)+1} \times A^{s(n)+1} &\to \{0,1\}
\end{aligned}
$$

The predicates

$$\forall c \in A^{s(n)+1} : \text{legal}(c) = 1 \Leftrightarrow c \text{ is a legal configuration of } M$$

$$\forall c \in A^{s(n)+1} : \text{start}(c) = 1 \Leftrightarrow c \text{ is the start configuration of } M \text{ on input } w$$

$$\forall c \in A^{s(n)+1} : \text{accept}(c) = 1 \Leftrightarrow c \text{ is an accepting configuration}$$

$$\forall (c_1, c_2) \in A^{s(n)+1} \times A^{s(n)+1} : \text{succ}(c_1, c_2) = 1 \Leftrightarrow c_1 \text{ yields } c_2$$

The predicates and the language $L(M)$

Observation

$$w \in L(M) \Leftrightarrow \exists c_1, \ldots, c_{t(n)} \in A^{s(n)+1} :$$

$$\bigwedge_{i=1}^{t(n)} \mathrm{legal}(c_i) \wedge \mathrm{start}(c_1) \wedge \mathrm{accept}(c_{t(n)}) \wedge \bigwedge_{i=1}^{t(n)-1} \mathrm{succ}(c_i, c_{i+1}).$$

Replacing the predicates by Boolean formulas

The variables

Variables

$$x_{i,j,s}, 1 \leq i \leq t(n), 1 \leq j \leq s(n) + 1, s \in A,$$

such that

$$x_{i,j,s} = 1 \text{ iff the } j\text{-th symbol in configuration } c_i \text{ is } s$$

The formula for legal

$$\phi_{\text{legal}} = \bigwedge_{\substack{1 \leq i \leq t(n) \\ 1 \leq j \leq s(n)}} \left[\left(\bigvee_{s \in A} x_{i,j,s} \right) \wedge \left(\bigwedge_{\substack{s,t \in A \\ s \neq t}} (\bar{x}_{i,j,s} \vee \bar{x}_{i,j,t}) \right) \right]$$

Replacing the predicates by Boolean formulas

The formula for start

$$\phi_{\text{start}} = x_{1,1,q_0} \wedge x_{1,2,\rhd} \wedge$$
$$x_{1,3,w_1} \wedge \cdots \wedge x_{1,n+2,w_n} \wedge$$
$$x_{1,n+3,\sqcup} \wedge \cdots \wedge x_{1,s(n)+1,\sqcup}$$

The formula for accept

$$\phi_{\text{accept}} = \bigvee_{\substack{1 \le i \le t(n) \\ 1 \le j \le s(n)}} x_{i,j,q_{\text{accept}}}$$

Replacing the predicates by Boolean formulas

Windows

- We call the 2×3 window consisting of symbols in positions $j-1, j, j+1$ in configurations c_i, c_{i+1} the (i,j)-th window
- a window is called legal if it does not violate the actions specified by M's transition function δ
- legal windows

$$\bigvee_{\substack{a_1,\ldots,a_6 \\ \text{is a legal window}}} (x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge \cdots \wedge x_{i+1,j+1,a_6})$$

The formula for succ

$$\phi_{\text{succ}} = \bigwedge_{\substack{1 \leq i \leq t(n)-1 \\ 2 \leq j \leq s(n)}} \text{the } (i,j)\text{-th window is legal}$$

Completing the proof for Theorem 2.10

- $\phi := \phi_{\text{legal}} \wedge \phi_{\text{start}} \wedge \phi_{\text{succ}} \wedge \phi_{\text{accept}}$
- $w \in L(M) \Leftrightarrow \phi \in SAT$.
- If M is a polynomial time Turing machine, then there is a $k \in \mathbb{N}$ such that for all $n \in \mathbb{N}$ $t(n), s(n) \leq n^k$.
- In that case, on input w the formula ϕ can be constructed in time polynomial in $|w|$.

The problem for **PSPACE** and *TQBF*

Problem and hint for solution

- If TM M is only polynomial space n^k, the best we know is that is has run time $2^{\mathcal{O}(n^k)}$.
- But did not use quantifiers (more precisely, only used existential quantifiers).
- Extend successor predicate by using quantifiers.

Extended successor predicate and $L(M)$

Extended successor predicate succ_l

$$\forall (c_1, c_2) \in A^{s(n)+1} \times A^{s(n)+1} : \mathrm{succ}_l(c_1, c_2) = 1 \Leftrightarrow c_2 \text{ is reachable from } c_1 \text{ with at most } 2^l \text{ steps of } M$$

Observations

- For $l := \lceil \log(t(n)) \rceil$:

$$w \in L(M) \Leftrightarrow \exists c_1, c_2 \in A^{s(n)+1} : \mathrm{start}(c_1) \wedge \mathrm{accept}(c_2) \wedge \mathrm{succ}_l(c_1, c_2)$$

- $\mathrm{succ}_l(c_1, c_2) \Leftrightarrow \exists c_3 : \mathrm{legal}(c_3) \wedge \mathrm{succ}_{l-1}(c_1, c_3) \wedge \mathrm{succ}_{l-1}(c_3, c_2)$

An auxilliary predicate for succ_l

Auxilliary predicate H

$H : \left(A^{s(n)+1}\right)^5 \to \{0,1\}$, with

$$H(c_1, \ldots, c_5) = \neg\Big(\big((c_1, c_3) = (c_4, c_5)\big) \vee \big((c_3, c_2) = (c_4, c_5)\big)\Big).$$

A short description for succ_l

$$\mathrm{succ}_l(c_1, c_2) \Leftrightarrow \exists c_3 \forall c_4 \forall c_5 : \\ \mathrm{legal}(c_3) \wedge \big(H(c_1, \ldots, c_5) \vee \mathrm{succ}_{l-1}(c_4, c_5)\big).$$

Completing the proof for Theorem 2.11 (1)

- M a polynomial space TM, choose $k \in \mathbb{N}$ such that M has space complexity $s(n) = n^k$ and time complexity $t(n) = 2^{n^k}$. Set $l := n^k$.
- From definition of succ_l:

$$w \in L(M) \Leftrightarrow \exists c_1, c_2 \in A^{s(n)+1} : \\ \mathrm{start}(c_1) \wedge \mathrm{accept}(c_2) \wedge \mathrm{succ}_{n^k}(c_1, c_2).$$

- Replace succ_l by its short description to obtain

$$w \in L(M) \Leftrightarrow \exists c_1 \exists c_2 \exists c_3 \forall c_4 \forall c_5 \in A^{s(n)+1} : \\ \mathrm{start}(c_1) \wedge \mathrm{accept}(c_2) \wedge \\ \left(\mathrm{legal}(c_3) \wedge \left(H(c_1, \ldots, c_5) \vee \mathrm{succ}_{n^k-1}(c_4, c_5)\right)\right).$$

- Repeat this process with $\mathrm{succ}_{l-1}, \mathrm{succ}_{l-2}, \ldots, \mathrm{succ}_1$.

Completing the proof for Theorem 2.11 (2)

- Obtain
  $w \in L(M) \Leftrightarrow Q_1 c_1 Q_2 c_2 \ldots Q_B c_B \in A^{s(n)+1}\ \psi(c_1, \ldots, c_B)$,
  where
  1. $B = B(n)$ is polynomial in n
  2. $Q_j \in \{\exists, \forall\}, j = 1, \ldots, B$
  3. $\psi(\cdot)$ is a predicate of polynomial size using Boolean operators and the predicates $\mathrm{start}, \mathrm{accept}, \mathrm{legal}, \mathrm{succ}$.
- Use variables $x_{i,j,s}$ and Boolean predicates as before to obtain a fully quantified Boolean formula of size polynomial in $|w| = n$ that is true iff $w \in L(M)$.
- The formula can be computed in polynomial time.