2. Lower bounds

We turn to a complexity theory for computational problems.

Let $t: \mathbb{N} \to \mathbb{N}$ be a resource bound, and let A be any set:

- t is a lower bound for A w.r.t. Φ-complexity iff $A \notin \Phi(t)$
- t is an upper bound for A w.r.t. Φ-complexity iff $A \in \Phi(t)$

Remark:

No greatest lower bound if Φ admits linear compr. / speed-up:
Suppose $A \notin \Phi(t)$, t greatest lower bound for A.
Then, $A \notin \Phi(t)$ but $A \in \Phi(2t)$. By lin. compr. /speed-up,
$A \in \Phi(t)$ $\lightning$

Goal: Proving lower bounds for concrete problems

Methods:

- completeness methods (based on diagonalization)
- Counting method (based on the pigeonhole principle)

2.1 The completeness method

Idea: Comparing problems, i.e., prove statements like
A is comp. harder than B

2.1.1 Reducibility relations

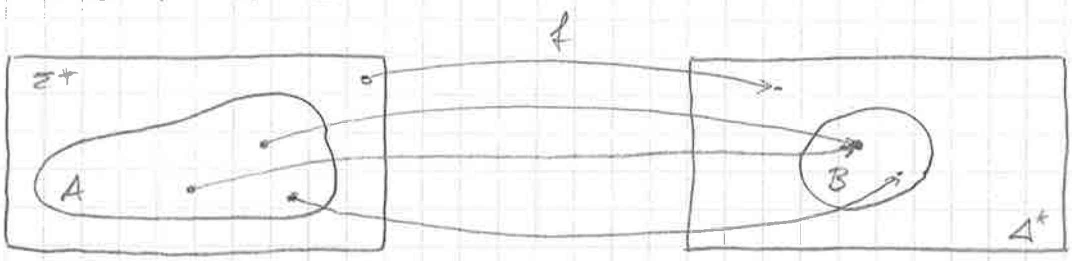
Definition 1.

Let $A \subseteq \Sigma^*$, $B \subseteq \Delta^*$ be languages.

(1) $A \leq_m^p B \Leftrightarrow_{def}$ there is an $f \in FP$ s.t. for all $x \in \Sigma^*$,
$$x \in A \Leftrightarrow f(x) \in B$$

(2) $A \leq_m^{log} B \Leftrightarrow_{def}$ there is an $f \in FL$ s.t. for all $x \in \Sigma^*$,
$$x \in A \Leftrightarrow f(x) \in B$$

Reduction principle:



Example: Consider foll. problems:

$$\text{SUBSET SUM} =_{def} \left\{ (a_1, \dots, a_m, b) \mid (\exists I \subseteq \{1, \dots, m\}) \left[\sum_{i \in I} a_i = b \right] \right\}$$

$$\text{PARTITION} =_{def} \left\{ (a_1, \dots, a_m) \mid (\exists I \subseteq \{1, \dots, m\}) \left[\sum_{i \in I} a_i = \sum_{i \notin I} a_i \right] \right\}$$

We show SUBSET SUM $\leq_m^{log}$ PARTITION.

Define $f : (a_1, \dots, a_m, b) \mapsto (a_1, \dots, a_m, b+1, N-b+1)$
where $N =_{def} \sum_{i=1}^{m} a_i$.

So, $f(a_1, \dots, a_m, b) = (a_1', \dots, a_{m+2}')$ s.t.
$a_i' = a_i$ for $i \in \{1, \dots, m\}$, $a_{m+1}' = b+1$, $a_{m+2}' = N-b+1$

Claim: $(a_1, \dots, a_m, b) \in$ SUBSET SUM $\iff f(a_1, \dots, a_m, b) \in$ PARTITION

$\boxed{\Rightarrow}$ Let $(a_1, \dots, a_m, b) \in$ SUBSET SUM, i.e., there is an $I \subseteq \{1, \dots, m\}$ s.t. $\sum_{i \in I} a_i = b$. Define $I' =_{def} I \cup \{m+2\}$. Then,

$$\sum_{i \in I'} a_i' = \sum_{i \in I} a_i' + a_{m+2}' = \underbrace{\sum_{i \in I} a_i}_{= b} + N - b + 1 = N + 1$$

$$\sum_{i \notin I'} a_i' = \sum_{i \notin I} a_i' + a_{m+1}' = \underbrace{\sum_{i \notin I} a_i}_{= N-b} + b + 1 = N + 1$$

Hence, $f(a_1, \dots, a_m, b) \in$ PARTITION

$\boxed{\Leftarrow}$ Let $f(a_1, \dots, a_m, b) \in$ PARTITION, i.e., there is an $I' \subseteq \{1, \dots, m+2\}$ s.t.

$$\sum_{i \in I'} a_i' = \sum_{i \notin I'} a_i' = \frac{1}{2} \cdot \sum_{i=1}^{m+2} a_i' = N + 1$$

It holds: $m+2 \in I' \iff m+1 \notin I'$ (since $a_{m+1}' + a_{m+2}' = N+2$)

W.l.o.g. assume $m+2 \in I'$. Define $I =_{def} I' \setminus \{m+2\}$.

Then,

$$\sum_{i \in I} a_i = \sum_{i \in I'} a_i' - a_{m+2}' = N + 1 - (N - b + 1) = b.$$

Hence, $(a_1, \dots, a_m, b) \in$ SUBSET SUM

Running space of TM summing up all a_i's to comp. N is $O(\log n)$.
That is, $f \in FL$.

Proposition 2.

(1.) $A \le_m^{log} B \implies A \le_m^p B$

(2.) $\le_m^p$, $\le_m^{log}$ are reflexive and transitiv.

(3.) $A \in P$, $B, \overline{B} \neq \emptyset \implies A \le_m^p B$

(4.) $A \in L$, $B, \overline{B} \neq \emptyset \implies A \le_m^{log} B$

Proof:

(1): Clear.

(2): Exercise.

(3): Let $A \in P$, $B, \overline{B} \neq \emptyset$, i.e., there exist $x_1 \in B$, $x_2 \in \overline{B}$.

Define

$$f(x) =_{def} \begin{cases} x_1 & \text{if } x \in A \\ x_2 & \text{if } x \notin A \end{cases}$$

Then, $f \in FP$ and $x \in A \Leftrightarrow f(x) \in B$. Thus, $A \le_m^p B$.

(4): Analogous to (3) $\blacksquare$

Experience (not a theorem): $\le_m^p$-reductions can be replaced by $\le_m^{log}$; so, we consider only $\le_m^{log}$.

Closure of (complexity) class $\mathcal{K}$ under $\leq_m^\tau$:

- $\mathcal{R}_m^\tau(\mathcal{K}) =_{def} \{A \mid (\exists B \in \mathcal{K})[A \leq_m^\tau B]\}$
- $\mathcal{R}_m^\tau(B) =_{def} \mathcal{R}_m^\tau(\{B\}) = \{A \mid A \leq_m^\tau B\}$
- $\mathcal{K}$ is closed under $\leq_m^\tau$ $\Leftrightarrow_{def}$ $\mathcal{R}_m^\tau(\mathcal{K}) = \mathcal{K}$

Proposition 3.

(1.) $\mathcal{K} \subseteq \mathcal{R}_m^{\log}(\mathcal{K})$

(2.) $\mathcal{K} \subseteq \mathcal{K}' \Rightarrow \mathcal{R}_m^{\log}(\mathcal{K}) \subseteq \mathcal{R}_m^{\log}(\mathcal{K}')$

(3.) $\mathcal{R}_m^{\log}(\mathcal{R}_m^{\log}(\mathcal{K})) = \mathcal{R}_m^{\log}(\mathcal{K})$

In other words: $\mathcal{R}_m^{\log}$ is a hull operator.

Proof:

(1.) Follows from $A \leq_m^{\log} A$.

(2.) Clear.

(3.) It suffices to show $\mathcal{R}_m^{\log}(\mathcal{R}_m^{\log}(\mathcal{K})) = \mathcal{R}_m^{\log}(\mathcal{K})$. Let $A \in \mathcal{R}_m^{\log}(\mathcal{R}_m^{\log}(\mathcal{K}))$, i.e., $A \leq_m^{\log} B$ for some $B \in \mathcal{R}_m^{\log}(\mathcal{K})$. Then, there is a $C \in \mathcal{K}$ s.t. $B \leq_m^{\log} C$. By transitivity, we obtain $A \leq_m^{\log} C$. Thus, $A \in \mathcal{R}_m^{\log}(\mathcal{K})$ $\blacksquare$

Theorem 4.

Let $X \in \{D, N\}$, $s(n) \geq \log n$ be space-constructible, $t(n) \geq n$.

(1.) $\mathcal{R}_m^{\log}(XSPACE(s)) = XSPACE(s(\mathrm{Pol}\, n))$

(2.) $\mathcal{R}_m^{\log}(XTIME(\mathrm{Pol}\; t)) = XTIME(\mathrm{Pol}\; t(\mathrm{Pol}\, n))$

Proof: (only (1) for $X = D$)

$\boxed{\subseteq}$ Let $A \in \mathcal{R}_m^{\log}(DSPACE(s))$, i.e., there ex. $B \in DSPACE(s)$ s.t. $A \leq_m^{\log} B$ via $f \in FL$. Then, $x \in A \Leftrightarrow f(x) \in B$ and $|f(x)| \leq |x|^k$ for some $k > 0$.

Define M to be that TM that, on input x,

(1) computes $f(x)$ (in space $\log |x|$)

(2) computes $c_B(f(x))$ (in space $s\,|f(x)|$)

Hence, M accepts A in space $s(|x|^k)$.

Thus, $A \in DSPACE(s(n^k))$

$\boxed{\supseteq}$ Let $A \in DSPACE(s(n^k))$ for some $k \in \mathbb{N}_+$. We use padding: $A_{n^k} \in DSPACE(s)$.

We have to show: $A \leq_m^{\log} A_{n^k}$. Define $f: x \mapsto x b a^{|x|^k - x - 1}$

Then, $f \in FL$ and $x \in A \Leftrightarrow f(x) \in A_{n^k}$.

Hence, $A \in \mathcal{R}_m^{\log}(DSPACE(s))$. ∎

Corollary 5.

(1.) L, NL, P, NP, PSPACE, EXP, NEXP are closed under $\leq_m^{\log}$.

(2.) LIN, NLIN, E, NE are not closed under $\leq_m^{\log}$.

Proof: (examples)

(1.) $\mathcal{R}_m^{\log}(NL) = \mathcal{R}_m^{\log}(NSPACE(\log n)) \overset{\text{Thm } 4}{=} NSPACE(\log \mathrm{Pol}\, n)$

$= \bigcup_{k \in \mathbb{N}} NSPACE(\log n^k) = \bigcup_{k \in \mathbb{N}} NSPACE(k \cdot \log n)$

$= NL$

(2.) It holds that

$$\begin{aligned} PSPACE &= DSPACE(Pol\ n) \\ &\overset{\text{Thm 4}}{=} \mathcal{R}_m^{\log}(DSPACE(n)) \\ &= \mathcal{R}_m^{\log}(LIN) \\ &\subseteq \mathcal{R}_m^{\log}(NLIN) \\ &\overset{\text{Thm 4.}}{=} NSPACE(Pol\ n) \\ &= PSPACE \end{aligned}$$

Thus, $\mathcal{R}_m^{\log}(LIN) = \mathcal{R}_m^{\log}(NLIN) = PSPACE \quad \supseteq NLIN \supseteq LIN$

2.1.2 Complete problems

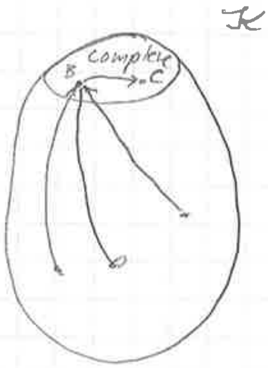
Definition 6.

Let $\mathcal{K}$ be closed under $\leq_m^\tau$ ($\tau \in \{\log, p\}$), and let B be any set.

(1.) B is hard for $\mathcal{K}$ w.r.t. $\leq_m^\tau$ $:\Leftrightarrow_{def}$ $\mathcal{K} \subseteq \mathcal{R}_m^\tau(B)$.

(2.) B is complete for $\mathcal{K}$ w.r.t. $\leq_m^\tau$ $:\Leftrightarrow_{def}$ $\mathcal{K} = \mathcal{R}_m^\tau(B)$.

We also say B is $\leq_m^\tau$-hard ($\leq_m^\tau$-complete) for $\mathcal{K}$.



Suppose B is $\leq_m^\tau$-complete for $\mathcal{K}$.
Now, let $C \in \mathcal{K}$ be another set s.t.
$B \leq_m^\tau C$. Then, C is $\leq_m^\tau$-complete for $\mathcal{K}$.

Proposition 7.

Let $\mathcal{K}_1, \mathcal{K}_2$ be closed under $\leq_m^\tau$, and let B be $\leq_m^\tau$-complete for $\mathcal{K}_1$. Then,

$$\mathcal{K}_1 \subseteq \mathcal{K}_2 \iff B \in \mathcal{K}_2$$

Proof:

$\boxed{\Rightarrow}$ Clear.

$\boxed{\Leftarrow}$ $\mathcal{K}_1 = \mathcal{R}_m^\tau(B) \overset{B \in \mathcal{K}_2}{\subseteq} \mathcal{R}_m^\tau(\mathcal{K}_2) = \mathcal{K}_2$. ∎

Corollary 8.

Let B be $\leq_m^{\log}$-complete for $\mathcal{K}$.

(1) If $\mathcal{K} = NL$ then: $L = NL \iff B \in L$

(2) If $\mathcal{K} = P$ then: $NL = P \iff B \in NL$

(3.) If $\mathcal{K} = NP$ then: $P = NP \iff B \in P$

(4.) If $\mathcal{K} = PSPACE$ then: $NP = PSPACE \iff B \in NP$

(5.) If $\mathcal{K} = coNP$ then: $NP = coNP \iff B \in NP$

Theorem 9.

There are $\leq_m^{\log}$-complete sets for NL, P, NP, $PSPACE$, EXP, $NEXP$.

Proof: (for NP) Define a language

$$U =_{def} \{ x\#v\#u \mid x, v, u \in \{0,1\}^*, v \text{ is an encoding of a T-NTM } M \text{ accepting } x \text{ in } |u| \text{ steps} \}$$

There is a T-NTM M_U accepting $x\#v\#u$ in time $c \cdot |u| \cdot (|v| + |x|) \leq c \cdot |x\#v\#u|^2$, i.e., $U \in NP$.

Let $A \in NP$, i.e., there ex. T-NTM M accepting A in time p (p polynomial). Define

$$f_M(x) =_{def} x \# \text{encoding of } M \# 1^{p(|x|)}$$

Then, $f_M \in FL$

Moreover,

$$x \in A \iff M \text{ accepts } x \text{ in } p(|x|) \text{ steps}$$
$$\iff f_M(x) \in U$$

Complete problems for NL:

Graph accessibility problem (GAP):

Input: directed graph $G=(V,E)$, vertices $u,v \in V$
Question: Is there a (u,v)-path in G?

Theorem 10.

GAP is $\leq_m^{log}$-complete for NL.

Proof: Assume a graph $G=(V,E)$, $u,v \in V$ are as follows:

$$\boxed{\square\ v_1 \# v_2 \# \cdots \# v_m \diamond (v_1, v_{i_1}) \ldots (v_1, v_{i_r})(v_2, \cdot) \cdots (v_2, \cdot) \cdots (v_m, \cdot) \square}$$

We have to examine two cond.

(i) GAP $\in$ NL: Clear.

(ii) GAP is $\leq_m^{log}$-hard for NL:

Let $A \in NL$, i.e., there ex. 2-T-NTM M accepting A in logarithmic space. We consider encodings of configurations

$$(\underset{\leq m^{\log|x|}}{\text{w.t. inscriptions}};\ \text{pos. of }\underset{\log|x|}{\text{w.t. head}};\ \text{pos. of }\underset{|x|}{\text{i.t. head}},\ \underset{k}{\text{state}})$$

There are at most $m^{\log|x|} \cdot \log|x| \cdot |x| \cdot k \leq c \cdot |x|^r$ conf.

Define $G_x =_{def} (V_x, E_x)$ where

$V_x =_{def}$ set of all conf. of M on input x

$E_x =_{def}$ set of all pairs (k_1, k_2) s.t. k_2 is successor of k_1 in one nondet. step

Clearly, $x \mapsto G_x$ is comp. in log space

Let k_{init}, k_{acc} be unique initial, accepting conf.

Then, $x \in A \Leftrightarrow M$ accepts x
$\Leftrightarrow M$ reaches k_{acc} from k_{init} on x
$\Leftrightarrow (G_x, k_{init}, k_{acc})$

So, $f: x \mapsto (G_x, k_{init}, k_{acc}) \in FL$ and $A \leq_m^{log}$ GAP.

Complete problems for P:

Circuit value problem CVP:

Input: logical circuit using $\{\wedge, \vee, \neg\}$-gates (of arbitrary fan-in), assignment z

Question: Does the circuit evaluate to 1?

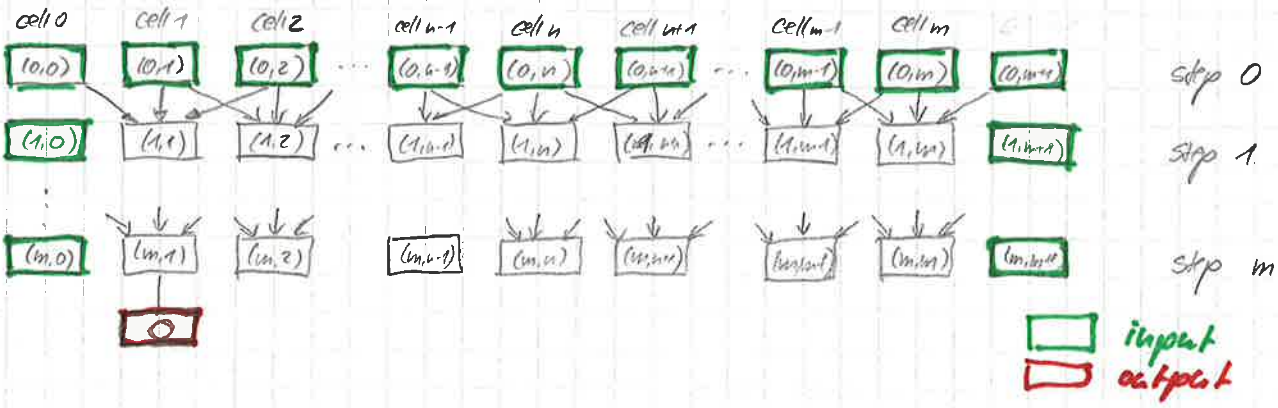
Theorem 11.

CVP is $\le_m^{\log}$-complete for P.

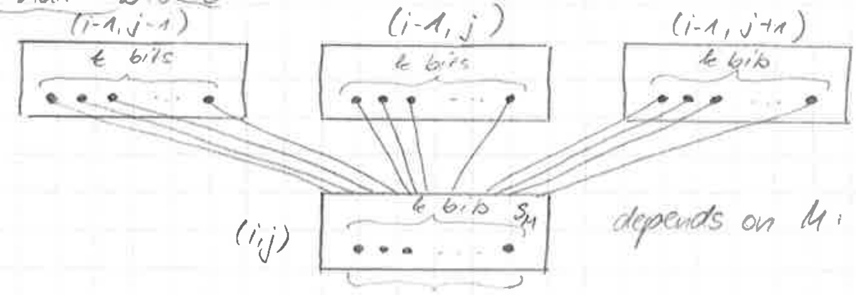
Proof: $CVP \in P$ is clear. We have to show: $A \in P \to A \le_m^{\log} CVP$.
Let $A \in P$, i.e., there ex. 1-TM M accepting A in time p (p polynomial). W.l.o.g. input x is given in cells $1, 2, \ldots, |x|$, during computation of M on x only cells $1, 2, \ldots, p(|x|)$ are used, M starts and halts in cell 1 with a clean tape.
Let S be the set of states, Σ alphabet, s_0 initial state, s_1 accepting halting state.
We construct on input x a logical circuit S_x (in log space) such that

$$x \in A \iff S_x \text{ evaluates to } 1$$
$$\iff S_x \in CVP$$

Main structure of S_x ($x = a_1 \ldots a_n$, $m = p(|x|)$):
block (i,j) contains encoding of symbol stored in cell j after step i (incl. state if head points to cell j):



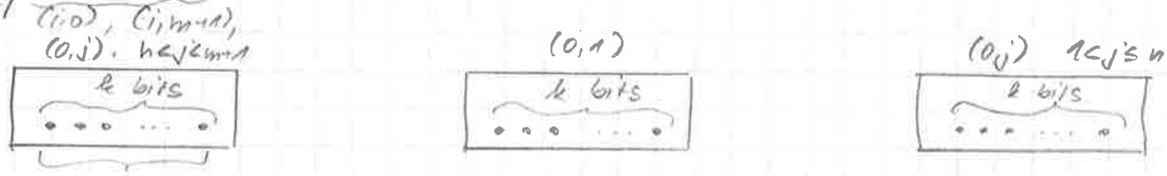
Internal blocks:



depends on M: circuit S_M using $\{\wedge,\vee,\neg\}$-gates

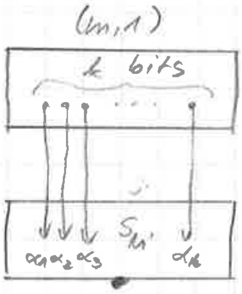
Code (symbol@cell j after step i)

Input blocks:



Code (□) code(a_1, s_0) code (a_j)

Output blocks:



code (□, state)

A depends on M: circuit S_M' using $\{\wedge,\vee,\neg\}$-gates

S_M' satisfies: $S_M'(\alpha_1,\dots,\alpha_k)=1 \iff$ state $= s_1$

We obtain:

$x \in A \iff M$ accepts x

$\iff M$ is in state s_1 and points to cell 1 containing symbol □ after $p(|x|)=m$ steps (on x)

$\iff S_x$ produces code (□, s_1) in block $(m,1)$

$\iff S_x$ produces 1 at output gate

$\iff S_x \in CVP$

$\iff f_M(x) \in CVP$ $\quad (f_M(x) =_{def} S_x)$

Furthermore, f_M is computable in log space.

Thus, $A \leq_m^{log}$ CVP.

Complete problems for NP:

Theorem 12.

Let $A \subseteq \Sigma^*$ be any language. Then,

$A \in NP \iff$ there ex. $B \in P$ and a polynomial p s.t.
for all $x \in \Sigma^*$,
$$x \in A \Leftrightarrow (\exists z)\left[|z| = p(|x|) \wedge (x,z) \in B \right]$$

Proof: Exercise. ∎

Circuit Satisfiability ? C-SAT:

Input: logical circuit C using $\{\wedge, \vee, \neg\}$-gates (of arbitrary fan-in)

Question: Is there an assignment z to the inputs of C s.t. $C(z)$ evaluates to 1?

Theorem 13.

C-SAT is $\leq_m^{log}$-complete for NP.

Proof: Containment: $C \in$ C-SAT $\iff (\exists z)\left[z \text{ ass. of } C \wedge (C,z) \in CVP \right]$.

Hardness: Let $A \in NP$, i.e., there ex. a $B \in P$, polynomial q s.t.

$x \in A \Leftrightarrow (\exists z)\left[|z| = q(|x|) \wedge (x,z) \in B \right]$

Let M be a T-TM accepting B on input $x\#z$ in time $p(|x\#z|)$.

That is,

$x \in A \iff (\exists z)\left[|z| = q(|x|) \wedge M \text{ accepts } x\#z \right]$

$\iff$ there ex. z s.t. $|z| = q(|x|)$ and $S_{x\#z}$ produces 1 at the output gate

$S_{x\#z}$ is the circuit constructed in the proof of theorem 11.

Define

$S'_x =_{\text{def}}$ circuit obtained from $S_{x\#u}$ by removing assignment u from input gates

Thus,

$x \in A \iff$ there ex. z s.t. $|z| = q(|x|)$ and $S_{x\#z}$ produces 1 at the output gate

$\iff$ there ex. z s.t. $|z| = q(|x|)$ and S'_x with assignment code(z) produces 1 at the output gate

$\iff S'_x \in$ C-SAT

Hence, $f_M : x \mapsto S'_x$ shows $x \in A \iff f_M(x) \in$ C-SAT.

Clearly, $f_M \in FL$. Thus, $A \leq_m^{\log}$ C-SAT. ∎

Satisfiability (SAT):

Input: prop. formula $H = H(x_1, .., x_n)$ over $\{\wedge, \vee, \neg\}$

Question: Is there a truth assignment to $x_1, .., x_n$ making H true?

3SAT:

Input: CNF $H = H(x_1, .., x_n)$ with exactly 3 literals in each clause

Question: Is there a truth assignment to $x_1, .., x_n$ making H true?

Theorem 14.

SAT and 3SAT are $\leq_m^{\log}$-complete for NP.

Proof:

- SAT, 3SAT $\in$ NP : clear.
- C-SAT $\leq_m^{\log}$ 3SAT (i.e., C-SAT $\leq_m^{\log}$ SAT):

Let S be a circuit with gates $v_1, \dots, v_r, v_{r+1}, \dots, v_s$, $v_1, \dots, v_r$ inputs, v_s output. Gate v_i computes a boolean function $f_i \in \{\wedge, \vee, \neg\}$, $i = r+1, \dots, s$.

Let v_{i_1}, v_{i_2} be predecessors of v_i ($i_1 = i_2$ if $v_i = \neg$).

x_i	x_{i_2}	x_{i_3}	$H_\wedge$	$H_\vee$
0	0	0	1	1
0	0	1	1	0
0	1	0	1	0
0	1	1	0	0
1	0	0	0	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

We have

$S \in$ C-SAT $\Longleftrightarrow$ $(\exists a_1, \dots, a_r \in \{0,1\})$[assigning $(a_1, \dots, a_r)$ to input gates of S yields 1 at output]

$\Longleftrightarrow$ $(\exists a_1, \dots, a_s \in \{0,1\})$[assigning $(a_1, \dots, a_r)$ to input gates of S yields a_i at gates v_i for $i \in \{r+1, \dots, s\}$ and $a_s = 1$]

$\Longleftrightarrow$ $(\exists a_1, \dots, a_s \in \{0,1\})\left[\bigwedge_{i=r+1}^{s} f_i(a_{i_1}, a_{i_2}) = a_i \wedge a_s = 1\right]$

$\Longleftrightarrow$ $H_S =_{def} \bigwedge_{i=r+1}^{s} H_i \wedge (x_s \vee x_s \vee x_s)$ satisfiable

$\Longleftrightarrow$ $H_S \in$ 3SAT

It remains to show how to transform f_i into 3CNF H_i:

- $f_i = \wedge$: $H_i =_{def} (x_i \leftrightarrow x_{i_1} \wedge x_{i_2})$

$$\equiv (x_i \vee \overline{x_{i_1}} \vee \overline{x_{i_2}}) \wedge (\overline{x_i} \vee x_{i_1} \vee x_{i_2}) \wedge (\overline{x_i} \vee x_{i_1} \vee \overline{x_{i_2}}) \wedge (\overline{x_i} \vee \overline{x_{i_1}} \vee x_{i_2})$$

- $f_i = \vee$: $H_i =_{def} (x_i \leftrightarrow x_{i_1} \vee x_{i_2})$

$$\equiv (x_i \vee x_{i_1} \vee \overline{x_{i_2}}) \wedge (x_i \vee \overline{x_{i_1}} \vee x_{i_2}) \wedge (x_i \vee \overline{x_{i_1}} \vee \overline{x_{i_2}}) \wedge (\overline{x_i} \vee x_{i_1} \vee x_{i_2})$$

- $f_i = \neg$: $H_i =_{def} (x_i \leftrightarrow \overline{x_{i_2}}) \equiv (x_i \vee x_i \vee x_{i_1}) \wedge (\overline{x_i} \vee \overline{x_i} \vee \overline{x_{i_1}})$

What is the simplest NP-complete SAT version to reduce from?

(k, l) - SAT:

Input: CNF $H = H(x_1, \dots, x_n)$ with exactly k literals in each clause such that each variable x_i occurs ~~in~~ exactly l ~~clauses~~ times as a literal
(Achtung: nicht in l Klauseln)

Question: Is there a truth assignment to $x_1, \dots, x_n$ making H true?

Fact:

(1) (k,l)-SAT $\leq_m^{\log} (k+1, l)$-SAT for $k, l \in \mathbb{N}_+$

(2) (k,l)-SAT $\leq_m^{\log} (k, l+1)$-SAT for $k, l \in \mathbb{N}_+$

(3) (k,l)-SAT is $\leq_m^{\log}$-complete for NP if $k \geq 3$ and $l \geq 4$; otherwise it is in P.

Tautology (TAUT):

Input: prop. formula $H = H(x_1, \dots, x_n)$

Question: Is H a tautology, i.e., is each truth assignment to $x_1, \dots, x_n$ a satisfying assignment for H?

Corollary 15.

TAUT is $\leq_m^{\log}$-complete for coNP.

Remark: 3SAT with each clause consisting of exactly 3 different literals in NP-Complete as well:

$$(x \vee y) \equiv (x \vee y \vee z) \wedge (\bar{z} \vee z' \vee z'') \wedge (\bar{z} \vee \bar{z}' \vee z'') \wedge (\bar{z} \vee z' \vee \bar{z}'') \wedge (\bar{z} \vee \bar{z}' \vee \bar{z}'')$$

Beyond NP.

Let Σ be an alphabet, $\|\Sigma\| \geq 2$. We define regular expressions over $\cup, \cdot, ^*$.

- $\emptyset$ is an expression
- If $a \in \Sigma$ then a is an expression.
- If H and H' are expressions then $H \cup H'$, $H \cdot H'$, H^* are expressions.

A regular expression H defines a language $L(H)$ according to the following rules:

- $L(\emptyset) =_{def} \emptyset$
- $L(a) =_{def} \{a\}$.
- $L(H \cup H') =_{def} L(H) \cup L(H')$.
- $L(H \cdot H') =_{def} \{ xy \mid x \in L(H), y \in L(H') \} = L(H) \cdot L(H')$
- $L(H^*) =_{def} L(H)^*$

We consider the following inequivalence problem f. reg. expr.:

$$INEQ(\Sigma, \cup, \cdot, ^*) =_{def} \{ (H, H') \mid L(H) \neq L(H') \}$$

We also discuss INEQ versions for reg. expr. defined by other operations, e.g., $INEQ(\Sigma, \cup, \cdot, ^2)$, $INEQ(\Sigma, \cup, \cdot, ^-)$

Theorem 16.

(1.) $INEQ(\Sigma, \cup, \cdot, ^*)$ is $\leq_m^{log}$-complete for PSPACE.

(2.) $INEQ(\Sigma, \cup, \cdot, ^2)$ is $\leq_m^{log}$-complete for NEXP.

(3.) $INEQ(\Sigma, \cup, \cdot, ^-)$ is $\leq_m^{log}$-hard for $DSPACE\left(2^{2^{\cdot^{\cdot^{2}}}}\Big\}^{O(\log n)}\right)$.

$\blacksquare$

2.1.3 Conditional lower bounds

Using hierarchy theorems we obtain certain strict lower bounds for complete problems:

(1.) $INEQ(\Sigma, \cup, \cdot, {}^*) \notin NSPACE(s)$ for monotone $s = o(n)$.

(2.) $INEQ(\Sigma, \cup, \cdot, {}^2) \notin NTIME(2^{c \cdot n})$ for some $c > 0$.

For interesting polynomial complexity classes we only obtain conditional lower bounds according to Cor. 8

Corollary 8':

let B be $\leq_m^{log}$-Complete for $\mathcal{K}$:

(1.) If $\mathcal{K} = NL$ then: $L \neq NL \Rightarrow B \notin L$

(2.) If $\mathcal{K} = P$ then: $NL \neq P \Rightarrow B \notin NL$

(3.) If $\mathcal{K} = NP$ then: $P \neq NP \Rightarrow B \notin P$

(4.) If $\mathcal{K} = PSPACE$ then: $NP \neq PSPACE \Rightarrow B \notin NP$

(5.) If $\mathcal{K} = coNP$ then: $NP \neq coNP \Rightarrow B \notin NP$

Corollary 8'':

(1.) If $L \neq NL$ then $GAP \notin L$

(2.) If $NL \neq P$ then $CVP \notin NL$

(3.) If $P \neq NP$ then $SAT \notin P$

(4.) If $NP \neq PSPACE$ then $INEQ(\Sigma, \cup, \cdot, {}^*) \notin NP$

(5.) If $NP \neq coNP$ then $TAUT \notin NP$

2.2 The counting method

Appropriate (but hard) for concrete computational models:

Idea: Let M be a TM that accepts A in Φ-complexity t.

- On different inputs $x \in A$, certain parameters $\gamma(x)$ observable during a run of M on x have to be different, otherwise M cannot make a distinction between diff. inputs
- For $\Phi(t)$-bounded comp., there are only $b(n)$ different $\gamma(x)$-values on input x (b injective, monotone)
- There are $a(n)$ different inputs $x \in A$ of length n
- Thus: $b(t(n)) \geq a(n)$ or $t(n) \geq b^{-1}(a(n))$

We consider $S =_{def} \{ ww^R \mid w \in \{0,1\}^* \}$

Theorem 17

$S \notin$ 1-T-DSPACE(s) for $s(n) = o(n)$.

Proof: Let M be a 1-T-DTM accepting S, let m be alphabet size of M, let k be the number of states of M. Let $u \in \{0,1\}^*$. Define

$\gamma(uu^R) =_{def}$ (state, pos. on w.t., tape inscriptions) when one-way reading head crosses the border between u and u^R

Consider u, v s.t. $|u| = |v|$ and $u \neq v$. It follows that $\gamma(uu^R) \neq \gamma(vv^R)$ (otherwise: uv^R is accepted by M).
We obtain:

(i) $\| \{ w \mid |w| = 2n,\ w \in S \} \| = \| \{ uu^R \mid |u| = n \} \| = \| \{ u \mid |u| = n \} \| = 2^n$

(ii) $\| \{ \gamma(uu^R) \mid |u| = n \} \| \leq \#$ conf. w. space $s(2n) \leq c^{s(2n)}$ for $c > 0$

Hence, $c^{s(2n)} \geq 2^n$, i.e., $s(2n) \geq d \cdot n$ for appr. $d > 0$.
Therefore, $s(n) \geq \frac{d}{2} n$ for infinitely many n ■

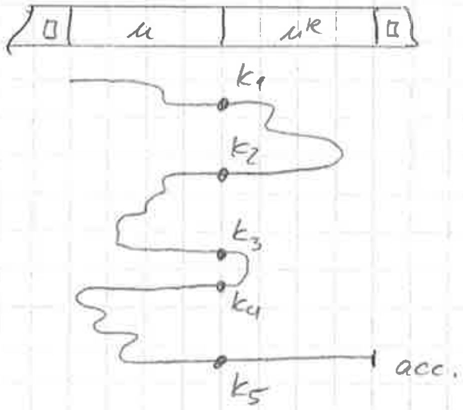
Theorem 18.

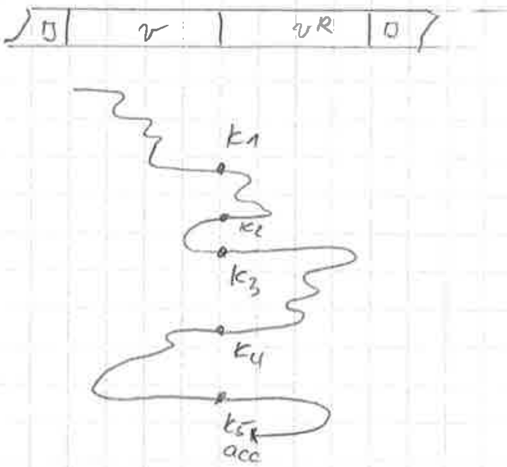
$S \notin$ 2-T-DSPACE(s) for $s(n) = o(\log n)$

Proof: let M be a 2-T-DTM accepting S, let m be the alphabet size of M, let k be the number of states. Let $u \in \{0,1\}^*$.
Define

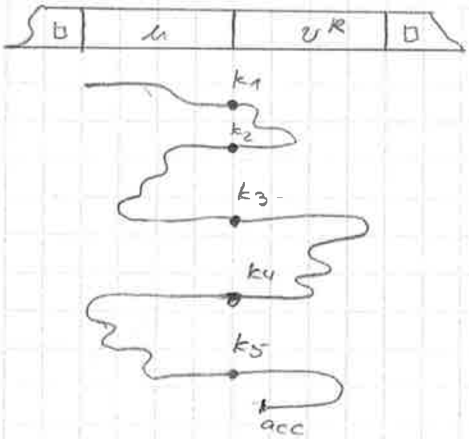
$\gamma(uu^R) =_{def}$ sequence of configurations
(state, position on w.t., tape inscr.)
„crossing sequence“ when crossing the border between u and u^R on the input tape

Consider u, v s.t. $|u| = |v| = n$ and $u \neq v$.
Assume $\gamma(uu^R) = \gamma(vv^R) = (k_1, k_2, \dots, k_r)$





Run of M on uv^R:



M accepts uv^R, but $uv^R \notin S$

$\rightarrow \gamma(uu^R) \neq \gamma(vv^R)$

(cut & paste!)

We obtain:

(i) $\| \{ w \mid |w| = 2n,\ w \in S \} \| = 2^n$

(ii) $$\begin{aligned} \| \{ \gamma(uu^R) \mid |u| = n \} \| &\le \sum_{r=0}^{R(n)} \left(\#\text{conf. w. space } s(2n)\right)^r \\ &\le \sum_{r=0}^{R(n)} \left(c^{s(2n)}\right)^r \\ &= \frac{\left(c^{s(2n)}\right)^{R(n)+1} - 1}{c^{s(2n)} - 1} \\ &\le c^{s(2n)(R(n)+1)} \qquad (*) \end{aligned}$$

It holds $R(n) \le 2 \cdot \#$ conf. w. space $s(2n) \le 2c^{s(2n)}$, since no conf. occurs twice in same direction of reading head. Thus,

$$\begin{aligned} \| \{ \gamma(uu^R) \mid |u| = n \} \| &\overset{(*)}{\le} c^{s(2n)\left(2c^{s(2n)}+1\right)} \\ &\le_{ae} 2^{2^{d \cdot s(2n)}} \qquad \text{for appr. } d > 0. \end{aligned}$$

Hence, $2^{2^{d s(2n)}} \ge 2^n$, i.e., $s(2n) \ge \frac{1}{d} \log n$.

Therefore, $s(n) \ge c' \cdot \log n$ for some $c' > 0$ and infinitely many n. ∎