18.404/6.840 Lecture 22

shrink me →

Last time:
- Finished NL = coNL
- Time and Space Hierarchy Theorems

Today:
- A "natural" intractable problem
- Oracles and P versus NP

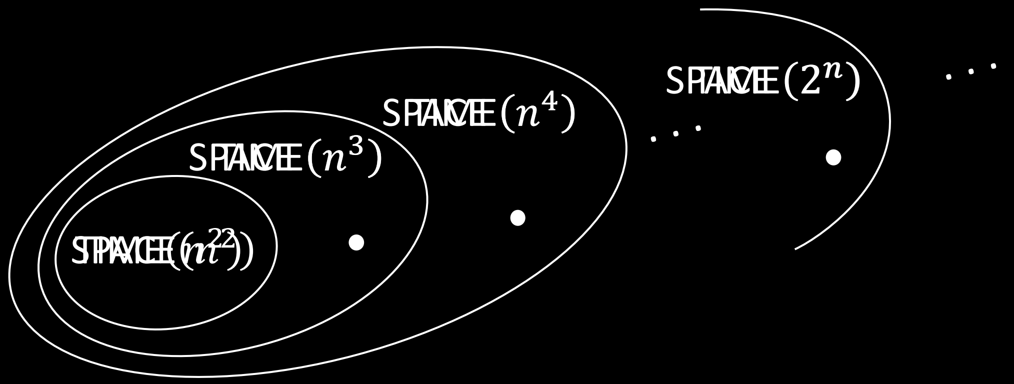
Posted:
- Solutions to Problem Set 5
- Problem Set 6

Review: Hierarchy Theorems

Theorems:

SPACE$\big(o(f(n))\big) \subsetneq$ SPACE$\big(f(n)\big)$ for space constructible f.

TIME$\Big(o\big(f(n)/\log(f(n))\big)\Big) \subsetneq$ TIME$\big(f(n)\big)$ for time constructible f.



Corollary: NL $\subsetneq$ PSPACE

Implies $TQBF \notin$ NL because the polynomial-time reductions in the proof that $TQBF$ is PSPACE-complete can be done in log space.

Check-in 22.1

Which of these are known to be true?
Check all that apply.

(a) TIME$(2^n) \subsetneq$ TIME(2^{n+1})

(b) TIME$(2^n) \subsetneq$ TIME(2^{2n})

(c) NTIME$(n^2) \subsetneq$ PSPACE

(d) NP $\subsetneq$ PSPACE

Check-in 22.1

Exponential Complexity Classes

Defn: EXPTIME = $\bigcup_k$ TIME $\left(2^{(n^k)}\right)$

EXPSPACE = $\bigcup_k$ SPACE $\left(2^{(n^k)}\right)$

Time Hierarchy Theorem

$\neq$

L $\subseteq$ NL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE $\subseteq$ EXPTIME $\subseteq$ EXPSPACE

$\neq$ $\neq$

Space Hierarchy Theorem

Defn: B is <u>EXPTIME-complete</u> if

1) $B \in$ EXPTIME
2) For all $A \in$ EXPTIME, $A \leq_{\mathrm{P}} B$

Same for <u>EXPSPACE-complete</u>

Theorem: If B is EXPTIME-complete then $B \notin$ P

Theorem: If B is EXPSPACE-complete then $B \notin$ PSPACE (and $B \notin$ P)

intractable

Next will exhibit an EXPSPACE-complete problem

A "Natural" Intractable Problem

Defn: $EQ_{\text{REX}} = \{\langle R_1, R_2 \rangle \mid R_1$ and R_2 are equivalent regular expressions$\}$

Theorem: $EQ_{\text{REX}} \in$ PSPACE
Proof: Later (if time) or exercise (uses Savitch's theorem).

Notation: If R is a regular expression write R^k to mean $\overbrace{RR\cdots R}^{k}$ (exponent is written in binary).
Defn: $EQ_{\text{REX}\uparrow} = \{\langle R_1, R_2 \rangle \mid R_1$ and R_2 are equivalent regular expressions with exponentiation$\}$

Theorem: $EQ_{\text{REX}\uparrow}$ is EXPSPACE-complete
Proof: 1) $EQ_{\text{REX}\uparrow} \in$ EXPSPACE
2) If $A \in$ EXPSPACE then $A \leq_{\text{P}} EQ_{\text{REX}\uparrow}$

1) Given regular expressions with exponentiation R_1 and R_2,
expand the exponentiation by using repeated concatenation and then use $EQ_{\text{REX}} \in$ PSPACE.
The expansion is exponentially larger, so gives an EXPSPACE algorithm for $EQ_{\text{REX}\uparrow}$.

2) Let $A \in$ EXPSPACE be decided by TM M in space $2^{(n^k)}$.

Give a polynomial-time reduction f mapping A to $EQ_{\text{REX}\uparrow}$.

Showing $A \leq_{\mathrm{P}} EQ_{\mathrm{REX}\uparrow}$

Theorem: $EQ_{\mathrm{REX}\uparrow}$ is EXPSPACE-complete

Proof continued: Let $A \in$ EXPSPACE decided by TM M in space $2^{(n^k)}$.

Give a polynomial-time reduction f mapping A to $EQ_{\mathrm{REX}\uparrow}$.

$$f(w) = \langle R_1, R_2 \rangle$$

$$w \in A \text{ iff } L(R_1) = L(R_2)$$

Construct R_1 so that $L(R_1) =$ all strings except a rejecting computation history for M on w.

Construct $R_2 = \Delta^*$ (Δ is the alphabet for computation histories, i.e., $\Delta = \Gamma \cup Q \cup \{\#\}$) ✓

R_1 construction: $R_1 = R_{\mathrm{bad-start}} \cup R_{\mathrm{bad-move}} \cup R_{\mathrm{bad-reject}}$

Rejecting computation history for M on w:

$$\underbrace{q_0 w_1 w_2 \cdots w_n \sqcup \cdots \sqcup}_{C_1 = C_{\mathrm{start}}} \# \underbrace{\text{ababa} \cdots \text{abababa}}_{C_2} \# \cdots \# \underbrace{\cdots q_{\mathrm{reject}} \cdots}_{C_{\mathrm{reject}}}$$

(each configuration has length $2^{(n^k)}$)

Check-in 22.2

Roughly estimate the size of the rejecting computation history for M on w.

(a) 2^n

(b) $2^{(n^k)}$

(c) $2^{2^{(n^k)}}$

Check-in 22.2

$A \leq_P EQ_{REX\uparrow}$ $(R_{\text{bad-start}})$

$\boxed{\text{Construct } R_1 \text{ to generate all strings except a rejecting computation history for } M \text{ on } w.}$

$R_1 = R_{\text{bad-start}} \cup R_{\text{bad-move}} \cup R_{\text{bad-reject}}$

Rejecting computation history for M on w:

$$\underbrace{q_0 w_1 w_2 \cdots w_n \sqcup \cdots \sqcup}_{C_1 = C_{\text{start}}}^{\longleftarrow 2^{(n^k)} \longrightarrow} \# \underbrace{\text{ababa} \cdots \text{abababa}}_{C_2}^{\longleftarrow 2^{(n^k)} \longrightarrow} \# \cdots \# \underbrace{\cdots q_{\text{reject}} \cdots}_{C_{\text{reject}}}^{\longleftarrow 2^{(n^k)} \longrightarrow}$$

$R_{\text{bad-start}}$ generates all strings that do not start with $C_{\text{start}} = q_0 w_1 w_2 \cdots w_n \sqcup \cdots \sqcup$

$R_{\text{bad-start}} = S_0 \cup S_1 \cup S_2 \cup \cdots \cup S_n \cup S_{\text{blanks}} \cup S_\#$

Remember: Δ is the alphabet for computation histories, i.e., $\Delta = \Gamma \cup Q \cup \{\#\}$)

Notation: $\Delta_\varepsilon = \Delta \cup \{\varepsilon\}$

$\Delta_{-\text{b}} = \Delta$ without b

Δ^7 = all strings of length 7

Δ_ε^7 = all strings of length 0 thru 7

$$S_{\text{blanks}} = \underbrace{\Delta^{n+1} \Delta_\varepsilon^{2^{(n^k)}-(n+2)}}_{\text{all strings of length } n+1 \text{ thru } 2^{(n^k)}-1} \Delta_{-\sqcup} \Delta^*$$

$S_0 = \Delta_{-q_0} \Delta^*$

$S_1 = \Delta \Delta_{-w_1} \Delta^*$

$S_2 = \Delta^2 \Delta_{-w_2} \Delta^*$

$\vdots$

$S_n = \Delta^n \Delta_{-w_n} \Delta^*$

$$\left\{ \begin{aligned} S_{n+1} &= \Delta^{n+1} \Delta_{-\sqcup} \Delta^* \\ &\vdots \\ S_{2^n-1} &= \Delta^{2^n-1} \Delta_{-\sqcup} \Delta^* \end{aligned} \right.$$

$S_\# = \Delta^{2^n} \Delta_{-\#} \Delta^*$

$A \le_{\mathrm{P}} EQ_{\mathrm{REX}\uparrow}$ $\left(R_{\text{bad−move}} \,\&\, R_{\text{bad−reject}}\right)$

Construct R_1 to generate all strings except a rejecting computation history for M on w.

$R_1 = R_{\text{bad−start}} \cup R_{\text{bad−move}} \cup R_{\text{bad−reject}}$

Rejecting computation history for M on w:

$$\underbrace{\overbrace{q_0 w_1 w_2 \cdots w_n \sqcup \;\cdots\; \sqcup}^{2^{(n^k)}} \#}_{C_1 \,=\, C_{\text{start}}} \underbrace{\overbrace{\text{ababa} \;\cdots\; \text{abababa}}^{2^{(n^k)}} \#}_{C_2} \qquad \cdots \qquad \# \underbrace{\overbrace{\cdots\; q_{\text{reject}} \;\cdots}^{2^{(n^k)}}}_{C_{\text{reject}}}$$

$R_{\text{bad−reject}}$ generates all strings that do not contain q_{reject}

$$R_{\text{bad−reject}} = \Delta^*_{-q_{\text{reject}}}$$

$R_{\text{bad−move}}$ generates all strings that contain an illegal 2×3 neighborhood

$$R_{\text{bad−move}} = \bigcup_{\text{illegal}} \left[\Delta^* \,\text{abc}\; \Delta^{2^{(n^k)}-2}\, \text{def}\; \Delta^*\right]$$

a	b	c
d	e	f

$$\cdots \quad \underbrace{\boxed{\text{abc} \overbrace{\qquad\qquad}^{2^{(n^k)}-2}}}_{C_i}\underbrace{\boxed{\text{def}}\qquad}_{C_{i+1}} \quad \cdots$$


Coffee
Break

Computation with Oracles

Let A be any language.
Defn: A TM M with oracle for A, written M^A, is a TM equipped with a "black box" that can answer queries "is $x \in A$?" for free.

Example: A TM with an oracle for SAT can decide all $B \in$ NP in polynomial time.

Defn: $\mathrm{P}^A = \{B \mid B$ is decidable in polynomial time with an oracle for $A\}$
Thus NP $\subseteq \mathrm{P}^{SAT}$
NP $= \mathrm{P}^{SAT}$? Probably No because coNP $\subseteq \mathrm{P}^{SAT}$

Defn: $\mathrm{NP}^A = \{B \mid B$ is decidable in nondeterministic polynomial time with an oracle for $A\}$
Recall $\mathit{MIN\text{-}FORMULA} = \{\langle\phi\rangle \mid \phi$ is a minimal Boolean formula $\}$
Example: $\overline{MIN-FORMULA} \in \mathrm{NP}^{SAT}$
"On input $\langle\phi\rangle$

1. Guess shorter formula ψ
2. Use SAT oracle to solve the coNP problem: ϕ and ψ are equivalent
3. *Accept* if ϕ and ψ are equivalent. *Reject* if not."

Oracles and P versus NP

Theorem: There is an oracle A where $\mathrm{P}^A = \mathrm{NP}^A$

Proof: Let $A = TQBF$

$\mathrm{NP}^{TQBF} \subseteq \mathrm{NPSPACE} = \mathrm{PSPACE} \subseteq \mathrm{P}^{TQBF}$

Relevance to the P versus NP question

Recall: We showed $EQ_{\mathrm{REX}\uparrow} \notin$ PSPACE.
Could we show $SAT \notin$ P using a similar method?

Reason: Suppose YES.
The Hierarchy Theorems are proved by a diagonalization.
In this diagonalization, the TM D simulates some TM M.
If both TMs were oracle TMs D^A and M^A with the same oracle A, the simulation and the diagonalization would still work.
Therefore, if we could prove P $\neq$ NP by a diagonalization, we would also prove that $\mathrm{P}^A \neq \mathrm{NP}^A$ for every oracle A.

But that is false!

Check-in 22.3

Which of these are known to be true?
Check all that apply.

(a) $\mathrm{P}^{SAT} = \mathrm{P}^{\overline{SAT}}$

(b) $\mathrm{NP}^{SAT} = \mathrm{coNP}^{SAT}$

(c) $\textit{MIN-FORMULA} \in \mathrm{P}^{TQBF}$

(d) $\mathrm{NP}^{TQBF} = \mathrm{coNP}^{TQBF}$

Check-in 22.3

Quick review of today

1. Defined EXPTIME and EXPSPACE
2. Defined EXPTIME- and EXPSPACE-completeness
3. Showed $EQ_{\mathrm{REX}\uparrow}$ is EXPSPACE-complete and thus $EQ_{\mathrm{REX}\uparrow} \notin$ PSPACE
4. Defined oracle TMs
5. Showed $\mathrm{P}^A = \mathrm{NP}^A$ for some oracle A
6. Discussed relevance to the P vs NP question

$EQ_{\mathrm{REX}} \in \mathrm{PSPACE}$

Theorem: $EQ_{\mathrm{REX}} \in \mathrm{PSPACE}$

Proof: Show $\overline{EQ_{\mathrm{REX}}} \in \mathrm{NPSPACE}$

"On input $\langle R_1, R_2 \rangle$ [assume alphabet Σ]

1. Convert R_1 and R_2 to equivalent NFAs N_1 and N_2 having m_1 and m_2 states.
2. Nondeterministically guess the symbols of a string s of length $2^{m_1 + m_2}$ and simulate N_1 and N_2 on s, storing only the current sets of states of N_1 and N_2.
3. If they ever disagree on acceptance then *accept*.
4. If always agree on acceptance then *reject*."