Time Complexity

Definition

Let $t : n \rightarrow n$ be a function.

- $TIME(t(n)) = \{L | L$ is a language decidable by a $O(t(n))$ deterministic TM$\}$
- $NTIME(t(n)) = \{L | L$ is a language decidable by a $O(t(n))$ non-deterministic TM$\}$

Polynomial Time

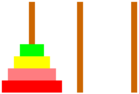
Definition

$$P = \bigcup_k TIME(n^k)$$

Example

$\{a^n b^n c^n | n \geq 0\} \in P$

Which are in P and Which arent?

Minimum Spanning Tree	The Towers of Hanoi	The Halting Problem
		
✓	✗	✗

Nondeterministic Polynomial Time

Definition

$$NP = \bigcup_k NTIME(n^k)$$

Example

the Traveling Salesman Problem (TSP) problem
the Integer Linear Programming (ILP) problem

Claim: $P \subseteq NP$
Proof: A deterministic Turing machine is a special case of non-deterministic Turing machines.

Exponential Time

Definition

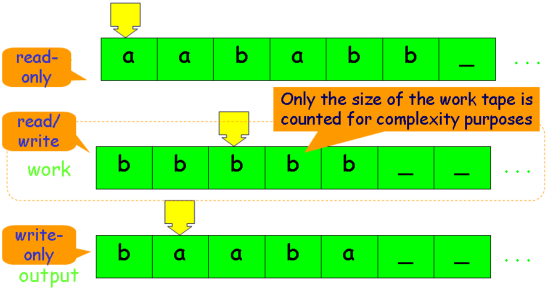
$$EXPTIME = \bigcup_k TIME(2^{n^k})$$

Definition

Let $s : n \rightarrow n$ be a function.

- $DSPACE(s(n)) = \{L | L$ is a language decidable by a $O(s(n))$ space deterministic TM$\}$
- $NSPACE(s(n)) = \{L | L$ is a language decidable by a $O(s(n))$ space non-deterministic TM$\}$

3-Tape TM



Logarithmic Space

Definition

$$L = DSPACE(log\ \ n)$$

$$NL = NSPACE(log\ \ n)$$

Polynomial Space

Definition

$$PSPACE = \bigcup_k DSPACE(n^k)$$

Example

$\{a^n b^n c^n | n \geq 0\} \in PSPACE$

Claim: $P \subseteq PSPACE$

Proof: A TM which runs in time $t(n)$ can use at most $t(n)$ space.

Observation

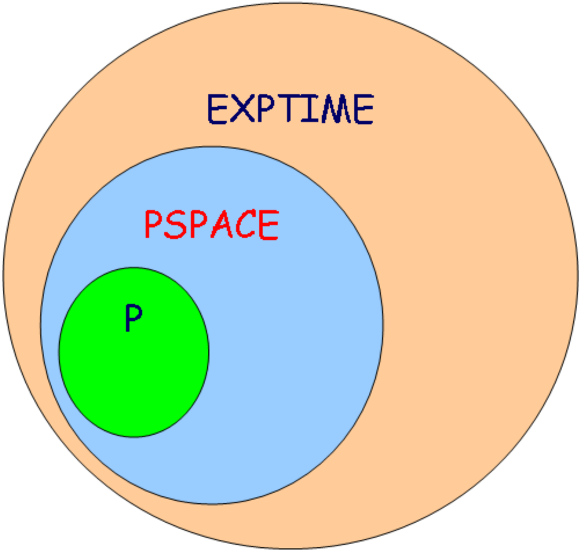
Theorem

$PSPACE \subseteq EXPTIME$

Proof.

A machine which uses polynomial space has at most exponential number of configurations (remember?). As deterministic machine that halts may not repeat a configuration, its running time is bounded by the number of possible configurations. □

Conjectured Relations Among Deterministic Classes



Two Important Theorems Regarding Space Complexity

Theorem (Savitch's Theorem)

$$\forall S(n) \geq log(n), NSPACE(S(n)) \subseteq SPACE(S(n)^2)$$

Theorem (Immerman's Theorem)

$$\forall S(n) \geq log(n), NSPACE(s(n)) = co - NSPACE(s(n))$$

Time and Space

- Recall: $TIME(f(n))$, $SPACE(s(n))$
- Questions:
  - how are these classes related to each other?
  - how do we define robust time and space classes?
  - what problems are contained in these classes? complete for these classes?

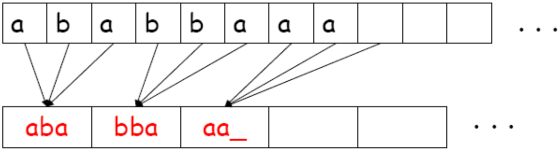
Linear Speedup

Theorem

Suppose TM M decides language L in time $f(n)$. Then for any $\epsilon > 0$, there exists TM M' that decides L in time $\epsilon \cdot f(n) + n + 2$.

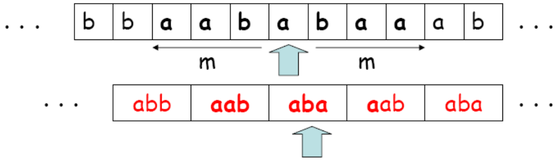
Proof Idea:

- compress input onto fresh tape:



Linear Speedup (cont'd)

- simulate M, m steps at a time



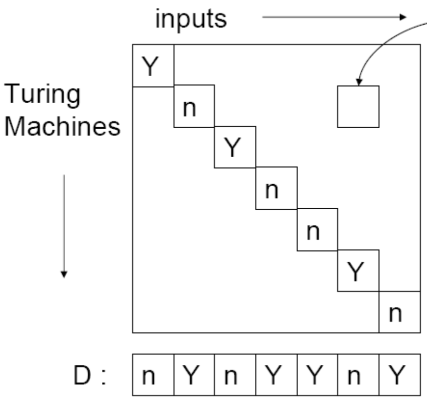
  – 4 (L,R,R,L) steps to read relevant symbols, "remember" in state

  – 2 (L,R or R,L) to make M's changes

- accounting:
  - part 1 (copying): $n+2$ steps
  - part 2 (simulation): $6(f(n)/m)$
  - set $m = 6/\epsilon$
  - total: $\epsilon \cdot f(n) + n + 2$

Hierarchy Theorems

- Does genuinely more time permit us to decide new languages?
- how can we construct a language L that is not in $TIME(f(n))$
- idea: same as "HALT undecidable" diagonalization and simulation



box (M, x): does M accept x in time f(n)?

- TM SIM tells us yes/no for each box **in time g(n)**
- rows include all of **TIME(f(n))**
- construct TM D running in time g(2n) that is not in table

Time Hierarchy Theorem

Theorem (Time Hierarchy Theorem)

For every proper complexity function $f(n) \geq n$,

$TIME(f(n)) \overset{\subset}{\neq} TIME(f(2n)^3)$.

Proper complexity function (also known as (fully) time-constructible function):

- $f(n) \geq f(n-1)$ for all n
- there exists a TM M that outputs exactly $f(n)$ symbols on input 1^n, and runs in time $O(f(n)+n)$ and space $O(f(n))$.
- includes all reasonable functions we will work with .
  $logn, \sqrt{n}, n^2, 2^n, n!, \ldots.$
  If f and g are proper then $f+g, fg, f(g), f^g, 2^g$ are all proper.
- can mostly ignore, but be aware it is a genuine concern.
- Theorem: $\exists$ non-proper f such that $TIME(f(n)) = TIME(2^f(n))$.

Proof of Time Hierarchy Theorem

- SIM is TM deciding language $\{< M, x >: M$ accepts x in $\leq f(|x|)$ steps $\}$
- Claim: SIM runs in time $g(n) = f(n)^3$.
- define new TM D: on input $< M >$
  - if SIM accepts $< M, M >$, reject
  - if SIM rejects $< M, M >$, accept.
- D runs in time $g(2n)$
- suppose M in $TIME(f(n))$ decides $L(D)$
  - $M(< M >) = SIM(< M, M >) \neq D(< M >)$
  - but $M(< M >) = D(< M >)$
- contradiction.

Space Hierarchy Theorem

Theorem (Time Hierarchy Theorem)

For every proper complexity function $f(n) \geq log_2 n$,

$DSPACE(f(n)) \overset{\subset}{\neq} DSPACE(f(n) log_2 n)$.

Robust Time and Space Classes

What is meant by "robust" class?

- no formal definition
- reasonable changes to model of computation shouldnt change class
- should allow "modular composition"
- calling subroutine in class (for classes closed under complement ...)

Examples:

$$L = DSPACE(logn)$$

$$PSPACE = \bigcup_k DSPACE(n^k)$$

$$P = \bigcup_k DTIME(n^k)$$

$$EXP = \bigcup_k DTIME(2^{n^k})$$

Relationships between classes

- How are these four classes related to each other?
- Time Hierarchy Theorem implies

$$P \overset{\subset}{\neq} EXP$$

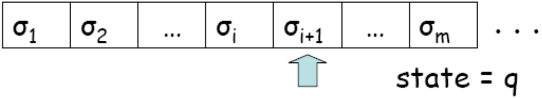
- Space Hierarchy Theorem implies

$$L \overset{\subset}{\neq} PSPACE$$

- L vs. P? $PSPACE$ vs. EXP?

Relationships between classes

- Useful convention: Turing Machine configurations. Any point in computation



represented by string:

$$C = \sigma_1\, \sigma_2 \ldots \sigma_i\, \mathbf{q}\, \sigma_{i+1}\, \sigma_{i+2} \ldots \sigma_m$$

- start configuration for single-tape TM on input x: $\mathbf{q_{start}} x_1 x_2 \ldots x_n$

Relationships between classes

- easy to tell if C **yields** C' in 1 step
- configuration graph: nodes are configurations, edge (C, C') iff C yields C' in one step
- # configurations for a 2-tape TM (work tape + read-only input) that runs in **space** t(n)

$$\underbrace{n}_{\text{input-tape head position}} \times \underbrace{t(n)}_{\text{work-tape head position}} \times \underbrace{|Q|}_{\text{state}} \times \underbrace{|\Sigma|^{t(n)}}_{\text{work-tape contents}}$$

Relationships between classes

- if $t(n) = c \log n$, at most

  $$n \times (c \log n) \times c_0 \times c_1^{c \log n} \leq n^k$$

  configurations.
- can determine if reach $\mathbf{q}_{\mathbf{accept}}$ or $\mathbf{q}_{\mathbf{reject}}$ from start configuration by exploring config. graph of size n^k (e.g. by DFS)
- Conclude: $\mathbf{L} \subset \mathbf{P}$

Relationships between classes

- if $t(n) = n^c$, at most

$$n \times n^c \times c_0 \times c_1{}^{n^c} \leq 2^{n^k}$$

configurations.

- can determine if reach $\mathbf{q}_{\mathbf{accept}}$ or $\mathbf{q}_{\mathbf{reject}}$ from start configuration by exploring config. graph of size 2^{n^k} (e.g. by DFS)

- Conclude: **PSPACE** $\subset$ **EXP**

- So far:

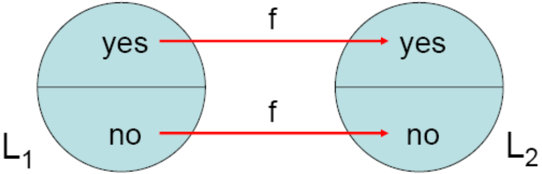
$$\mathbf{L} \subset \mathbf{P} \subset \mathbf{PSPACE} \subset \mathbf{EXP}$$

- believe all containments strict
- know $\mathbf{L} \subsetneq \mathbf{PSPACE}$, $\mathbf{P} \subsetneq \mathbf{EXP}$
- even before any mention of NP, two **major** unsolved problems:

$$\mathbf{L} \stackrel{?}{=} \mathbf{P} \qquad \mathbf{P} \stackrel{?}{=} \mathbf{PSPACE}$$

A P-complete problem

- We don't know how to prove $L \neq P$
- But, can identify problems in P least likely to be in L using P-completeness.
- need stronger reduction (why?)
- **logspace reduction**: f computable by DTM that uses $O(logn)$ space, denoted g$L_1 \leq_L L_2$
- If L_2 is P-complete, then L_2 in L implies $L = P$



A P-complete problem

Circuit Value (CVAL): given a variable-free Boolean circuit (gates $(\vee, \wedge, \neg, 0, 1)$, does it output 1?

Theorem

CVAL is P-complete.

CVAL is P-complete (proof)

- already argued in P
- L arbitrary language in P, TM M decides L in n^k steps

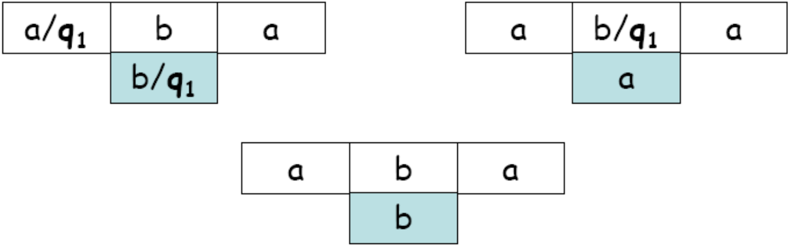
- Tableau (configurations written in an array) for machine M on input w:

w_1/q_s	w_2	...	w_n	...	_
w_1	w_2/q_1	...	w_n	...	_
w_1/q_1	a	...	w_n	...	_
		⋮			⋮
_/q_a	_	...	_	...	_

- height = time taken = $|w|^c$
- width = space used $\leq |w|^c$

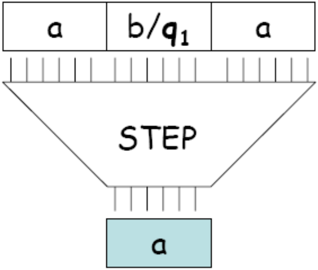
CVAL is P-complete (proof)

- Important observation: contents of cell in tableau determined by 3 others above it:



CVAL is P-complete (proof)

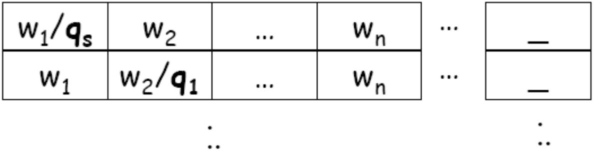
- Can build Boolean circuit STEP
  - input (binary encoding of) 3 cells
  - output (binary encoding of) 1 cell



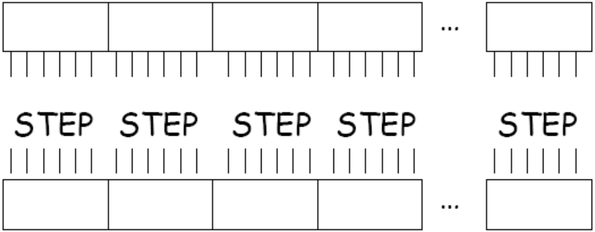
- each output bit is some function of inputs
- can build circuit for each
- size is independent of size of tableau

CVAL is P-complete (proof)

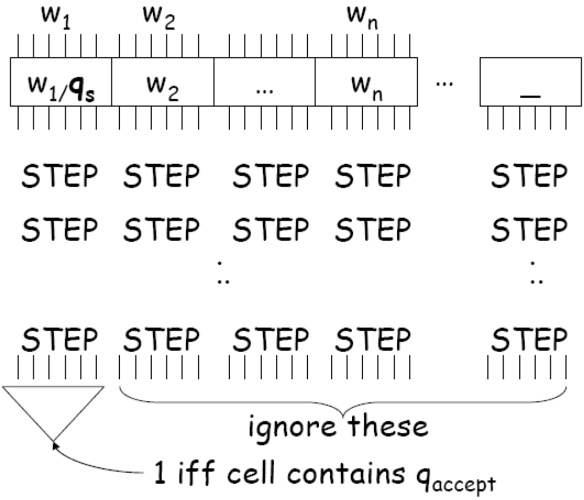
Tableau for M on input w



- $|w|^c$ copies of STEP compute row i from i-1



CVAL is P-complete (proof)



This circuit $C_{M,w}$ has inputs $w_1w_2\ldots w_n$ and $C(w) = 1$ iff M accepts input w.

logspace reduction

Size = $O(|w|^{2c})$

Summary

- First separations (via simulation and diagonalization):
  $P \neq EXP, \quad L \neq PSPACE$
- First major open questions:
  $L \stackrel{?}{=} P, \quad P \stackrel{?}{=} PSPACE$
- First complete problems: CVAL is P-complete