PROPOSITION 7 $NP \subseteq PSPACE$

Proof - Reduce from CNFSAT.
- Check all truth value assignments in lexicographic order (linear space in size of CNF formula)
- Accept if satisfying assignment found.
- Otherwise reject once all assignments tested.

OPEN PROBLEM 8 $NP \neq \mathcal{L}$?

EXAMPLE 9

Let $PATH = \{ \langle G, s, t \rangle \mid \exists$ path $s \rightsquigarrow t$ in digraph $G \}$
$PATH \in NL$

Proof If there is a path, there is one of length $\leq n = |V(G)|$
Keep counter $[1 .. n]$ - $\log n$ bits
Walk nondeterministically (guess next vertex and check on input tape that this is OK).
Accept if reached t before counter exceeded n
[vertex indices also require space $O(\log n)$.] ☐

Is PATH in $\mathcal{L}$? Excellent question

Would imply $\mathcal{L} = NL$ (i.e., PATH is NL-complete, will be discussed later.

Interestingly [Reingold '05] proved that UNDIRECTED PATH is in $\mathcal{L}$ ☐

THEOREM 10 SPACE HIERARCHY THEOREM

[Stearns, Hartmanis & Lewis '65] ← That is, before Cook-Levin

If f, g are space-constructible functions s.t. $f(n) = o(g(n))$ then

$$SPACE(f(n)) \subsetneq SPACE(g(n))$$

Proof Will skip this. Might be good exercise.

DEF 11 PSPACE-COMPLETENESS

L' is PSPACE-hard if $L \leq_p L'$ for every $L \in PSPACE$. If in addition $L' \in PSPACE$ then L' is PSPACE-complete.

A (not so interesting) PSPACE-complete language

$$SPACE\,BOUNDED\,TM = \{\langle M, x, 1^n\rangle \mid M \text{ accepts } x \text{ in space } n\}$$

Proof Problem set 2.

Let's look at a more interesting problem

DEF 12 A quantified Boolean formula (QBF) is a formula on the form

$$\varphi = Q_1 x_1\, Q_2 x_2 \cdots Q_n x_n\, \psi(x_1, \ldots, x_n)$$

where $Q_i \in \{\forall, \exists\}$

x_i ranges over $\{0,1\}$

ψ is a CNF formula

(not necessary, and Arora-Barak don't require this)

PRENEX NORMAL FORM: all quantifiers to the left.

Can easily convert to prenex.
Can easily convert to CNF / 3-CNF.

Note QBFs have determined truth value – either true or false.

Example 13

$\forall x \exists y \; (x \wedge y) \vee (\bar{x} \wedge \bar{y})$

"for all x exists y s.t. $x = y$" — true

$\forall x \forall y \; (x \wedge y) \vee (\bar{x} \wedge \bar{y})$

"for all x and all y they are always equal" – false

SAT – QBF with all quantifiers $\exists$

UNSAT – QBF – " – $\forall$ (and negated CNF inside)

THM 14 [Stockmeyer & Meyer '73]

The language

$$TQBF = \{ \psi \mid \psi \text{ is a true QBF} \}$$

is PSPACE-complete

Proof ~~FBQ~~ $TQBF \in PSPACE$ (sketch)

Let $\psi = Q_1 x_1 Q_2 x_2 \ldots Q_n x_n \; \varphi(x_1, \ldots, x_n)$ $|\varphi| = m$

Base case: If all variable set to values, just evaluate φ in $O(m)$ time and space

Inductive step

$\forall x_i \; \psi'$

set $x_i = 0$, evaluate, save — REUSING SPACE

set $x_i = 1$, evaluate, save

$\forall x_i \; \psi'$ true iff both values true

$O(1)$ extra space

$\exists x_i \; \psi'$

Similar, just check if one of $x_i = 0$ and $x_i = 1$ yields true value.

Total space usage something like $O(m+n)$

$L \in PSPACE \Rightarrow L \leq_p TQBF$

M decides L in space $s(n)$

Want to construct QBF ψ of size $O(s(n)^2)$

s.t. ψ true $\iff$ M accepts x

Let $m = k \cdot s(n)$ = # bits needed to encode config of M on input ~~G~~ x.

By Claim 5.3, $\exists$ CNF $\varphi_{M,x}$ s.t. for $C, C' \in \{0,1\}^m$ $\varphi_{M,x}(C, C') = 1$ if C and C' adjacent TM configs.

Use $\varphi_{M,x}$ to define ψ s.t. $\psi(C, C') = 1$ iff

$\exists$ path $C \rightsquigarrow C'$ in G_{M_x}.

Plug in C_{start} and C_{accept} $\Rightarrow$ Done!

Inductive definition

$\psi_i(C, C')$ = 1 iff $\exists$ path $C \rightsquigarrow C'$ of length $\leq 2^i$

$\psi_0 = \psi_{M,x}$

After $O(m)$ steps, get $\psi = \psi_{O(m)}$.

ATTEMPT 1

If $\exists$ path of length 2^i, then $\exists$ midpoint C'' s.t.

$$\psi_{i-1}(C, C'') \wedge \psi(C'', C')$$

Why not

$$\psi_i(C, C') := \exists C'' \; \psi_{i-1}(C, C'') \wedge \psi_{i-1}(C'', C')\,?$$

Not good: size doubles at each step $\Rightarrow$ exponential blow-up.

Need poly-size formula!

ATTEMPT 2

(these are collections of m variables each)

$$\psi_i(C, C') = \exists C'' \forall D^1 \forall D^2 \Big((D^1 = C \wedge D^2 = C'') \vee (D^1 = C'' \wedge D^2 = C') \Rightarrow \psi_{i-1}(D^1, D^2) \Big)$$

[$=$ and $\Rightarrow$ are just convenient shorthands. Can convert to CNF and prenex without problems]

"There is a midpoint C'' s.t. whenever D^1 is the starting point C and D^2 is the midpoint or D^1 is the midpoint and D^2 is the endpoint C', then there is a path from D^1 to D^2 in length $\leq 2^{i-1}$. The rest is just details..." ∎

A funny observation

Proof of Thm 14 established that anything in PSPACE reduces to TQBF via analysis of $G_{M,x}$.

But we never used out-degree 2 restriction...

So... $G_{M,x}$ could have been graph for NDTM M.

So... TQBF is NPSPACE-hard.

COROLLARY 15

$$PSPACE = NSPACE.$$

Can actually prove sth slightly more precise

THEOREM 16 (SAVITCH'S THEOREM '70)

For any space-constructible $s(n) \geq \log n$

$NSPACE(s(n)) \subseteq SPACE(s(n)^2)$.

Proof sketch Implement reduction in Thm 14 as recursive top-down procedure

Start with upper bound $2^{O(s(n))}$.

Check for all vertices in $G_{M,x}$ if can be midpoint $O(s(n))$ space. Recurse

$O(s(n))$ space per recursive call + $O(s(n))$ recursive calls + space reuse $\Rightarrow$ space $O(s(n)^2)$ ∎

PSPACE: optimal strategies for game-playing

View QBF as game

$$\exists x_1 \forall x_2 \exists x_3 \forall x_4 \cdots \varphi(x_1, x_2, x_3, x_4, \ldots)$$

$\exists$-player chooses value of x_1 s.t. for any choice by $\forall$-player for x_2 there is a choice by $\exists$-player for x_3 ...

... such that φ evaluates to true.

Can model other 2-player games with perfect information in this way.

Many such games PSPACE-complete.

Hard to see how winning strategy for 1st player could have concise description covering all 2nd player moves.

I.e., we are arguing that it seems

$$NP \neq PSPACE.$$

NEXT

- Computing in logarithmic space, and
- on the other side of NP (polynomial hierarchy again)

DD2445 COMPLEXITY THEORY

LECTURE 9½

Recap

- Space-bounded computation
- Polynomial space PSPACE ($\supseteq$ NP $\cup$ coNP)
- TQBF PSPACE-complete
  (proof used configuration graph $G_{M,x}$)
- PSPACE = NPSPACE
- Savitch's theorem: Nondeterminism can be simulated with quadratic blow-up in space

Next on the agenda

- Sublinear regime: logarithmic space L & NL
- NL-complete problems
- coNL
- ... After which we will move to the other side of NP again and talk about the polynomial hierarchy again [which we had to do somewhat prematurely before]

When studying logarithmic space and reducing between problems, polynomial-time reductions are no good

So powerful that the reduction can solve the problem

Clearly, we don't want reduction to be more powerful than actual algorithm
Hence, let us insist on reductions in logarithmic space.

Ok, good, but...
How can a log-space reduction compute polynomial-size output?

Two solutions:

(1) Write-only output tape on which space doesn't count.
Write once — write and move right
Never read; never move left

(2) Compute reduction bit by bit

Get equivalent definitions (good exercise to show)
We go for option (2)

DEF 1

$f: \{0,1\}^* \to \{0,1\}^*$ *implicitly logspace computable* if

a) f polynomially bounded $(\exists c \forall x \; |f(x)| \leq c \cdot |x|^c)$

b) $\mathcal{L}_f = \{ \langle x, i \rangle \mid f(x)_i = 1 \}$
$\mathcal{L}'_f = \{ \langle x, i \rangle \mid i \leq |f(x)| \}$ are both in $\mathcal{L}$

Language B is *logspace reducible* to language C, denoted $B \leq_\ell C$ if $\exists$ implicitly logspace computable f s.t. $x \in B \iff f(x) \in C$

C is *NL-complete* if $C \in NL$ and $\forall B \in NL \quad B \leq_\ell C$.

PROPOSITION 2

1. $B \leq_\ell C$ and $C \leq_\ell D \Rightarrow B \leq_\ell D$
2. $B \leq_\ell C$ and $C \in \mathcal{L} \Rightarrow B \in \mathcal{L}$

Proof Not hard but needs a bit of care. See textbook.

THEOREM 3 PATH is NL-complete

Recall $PATH = \{ \langle G, s, t \rangle \mid \exists$ path $s \rightsquigarrow t$ in digraph $G \}$

Proof Argued PATH $\in NL$ last time.

Let B in NL decided by M in log space

Define $f(x)$ to be configuration graph $G_{M,x}$ together with $|C_{start}| = s$ and $t =$ Caccept.

Representation adjacency matrix

1 in position (C, C') if C, C' legal transition.

size $G_{M,x}$ has $\leq 2^{\text{space}}$ vertices. $2^{(\log)} =$ poly — OK.

computation given C, C', look up current state and tape contents and check that C' is one of two possible configs to follow from C.

Certificate-style definition of NL?

"For every $x \in B$ $\exists$ witness y
s.t. $M(x,y) = 1$ and M runs in log space"

Need to be careful!

Suppose x CNF-formula, y satisfying assignment
Let M look up clauses in x one by one
then look up assignments in y
check that every clause satisfied.
Proves that CNF-SAT $\in$ NL
Hence NP = NL (and P=NP) – great!

Fix: Make certificate read-once

DEF 4 Certificate-style definition of NL

C is in NL if exists deterministic TM M (verifier)
with
- read-only input tape
- read-once certificate tape [read or move right each step]
- read-write tapes with $O(\log |x|)$ space bound

s.t. $x \in C \iff \exists u \in \{0,1\}^{p(|x|)}$ s.t. $M(x,u) = 1$

(for some fixed poly p depending on C).

LEMMA 5 Definitions 4 gives exactly the same class NL

Proof Exercise (not hard but useful).

Complements of space-bounded complexity classes

$\overline{\text{PATH}} = \{\langle G, s, t\rangle \mid \text{No path } s \rightsquigarrow t \text{ in digraph } G\}$

$\overline{\text{PATH}}$ in coNL (since PATH $\in$ NL)

In fact, $\overline{\text{PATH}}$ coNL-complete (since PATH NL-complete).

Log space NDTM deciding PATH:

Just walk nondeterministically for $|V(G)|$ steps from s, reject if didn't reach t

Most computation ~~paths~~ branches might reject, but if $\exists$ path then one branch will find it.

Log space NDTM deciding $\overline{\text{PATH}}$

Walk nondet & accept if didn't reach t?

A non-starter...

How can you make sure all branches find path $s \rightsquigarrow t$ for a no instance of PATH?!

Obviously can't be done, right? Seems clear that NL $\neq$ coNL, right? Wrong.

THM 6 NL = coNL — Immerman '88, Szelepcsényi '87

Proof Show that $\overline{\text{PATH}} \in$ NL.

Same ideas yield stronger statement (which we will not prove)

COR 7 For every space constructible $s(n) \geq \log n$ it holds that NSPACE$(s(n))$ = coNSPACE$(s(n))$

Moral: Don't trust your intuition too much regarding "obvious" truths in computational complexity theory (P vs NP, anyone?)

Proof of Thm 6

Provide read-once certificate for NL-complete language PATH

Important Read-once access to certificate
But can scan graph G as many times as wanted
(but not store on work tape)

$R(i) := \{v \in V(G) \mid \exists \text{ path } s \rightsquigarrow v \text{ of length} \leq i\}$

$n = |V(G)|$ denote $V(G) = \{1, 2, \ldots, n\}$
denote $r_i = |R(i)|$

Want to certify $t \notin R(n)$

Starting point: Certifying $v \in R(i)$ easy

Give vertices in path $u_0 = s, u_1, u_2, \ldots, u_{i'} = v$ for $i' \leq i$

Verification
- read vertices one by one
- keep u_j and u_{j+1} in memory - log space
- keep j in memory - log space
- at each step, check $(u_j, u_{j+1}) \in E(G)$ by scanning input tape.
- check that j never exceeds i.

Let such a certificate be denoted

IsMember(v, i) - $v \in R(i)$

Use this to construct two other types of certificates

(A) MEMBERSHIPEXPANSION (i, r, r')

Assuming $|R(i-1)| = r'$,
prove that $|R(i)| = r$

(B) NOTMEMBER (v, i, r)

Assuming that $|R(i)| = r$
prove that $v \notin R(i)$.

Suppose we can build such read-once verifiable subcertificates. Then we're done!
We all know $R(0) = \{s\}$ and $|R(0)| = 1$
~~Suppose~~ Let $r_i = |R(i)|$.
Here is the certificate

MEMBERSHIPEXPANSION $(1, r_1, 1)$;
MEMBERSHIPEXPANSION $(2, r_2, r_1)$;
MEMBERSHIPEXPANSION $(3, r_3, r_2)$;
⋮
MEMBERSHIPEXPANSION (n, r_n, r_{n-1});
NOTMEMBER (t, n, r_n)

- check each line in read-once fashion.
- keep counter i and neighbourhood size r_i
  → $\log n$ space
- finally verify nonmembership certificate
- each expansion certificate for step j is verified using stored r_{j-1} → $\log n$ space

(B) NotMember (v, i, r)

Suppose $R(i) = \{u_1, u_2, \ldots, u_r\}$ $u_1 < u_2 < \ldots < u_r$

Let certificate be sorted list of u_j's with membership certificates

u_1 : IsMember (u_1, i)
u_2 : IsMember (u_2, i)
$\vdots$
u_r : IsMember (u_r, i)

Denote this by =: ListMembers (i, r)

Verification
- r is known
- go over list and read u_j
- for each u_j, check certificate of membership
- check $u_j > u_{j-1}$
- check $u_j \neq v$
- check # u_j-vertices $= r$

(A) MembershipExpansion (i, r, r')

Use auxiliary certificate NotMemberOrNeighbour (v, i, r')

Assuming $|R(i+1)| = r'$, prof that $v \notin R(i+1)$

Reuse ListMembers (i, r')

Verification Go over list and verify u_j's as above
For each u_j, check $u_j \neq v$
and that $(u_j, v) \notin E(G_i)$

Now we can write down

MEMBERSHIPEXPANSION (i, r, r')

as an ordered list of subcertificates for vertices $1, 2, \ldots, n$

If vertex $j \in R(i)$, the line for j is

j: ISMEMBER (j, i)

If vertex $j \notin R(i)$, the line for j is

j: NOTMEMBERORNEIGHBOUR $(j, i-1, r')$

which is = LISTMEMBERS $(i-1, r')$

Verification

- For each j, check correctness of membership or non-membership certificate
- Count total # members; check that sum is $= r$

This concludes the proof ▨

Summary so far during the course (excluding the polynomial hierarchy) which we will return to next

$$\mathcal{L} \subseteq NL \subseteq P \subseteq NP \subseteq PSPACE \subseteq EXP$$

Some inclusions **must** be strict, since

- $\mathcal{L} \subsetneq PSPACE$ (space hierarchy theorem)
- $P \subsetneq EXP$ (time hierarchy theorem)

But we don't know which...
Probably most of them, or even all