LECTURE 4

News

- Pset 1 posted. Due Tue Sep 17 at 10am.
- Please read info about pset rules and peer evaluation process carefully
- Anything unclear? ⇒ Post a (private) question at Piazza

So far

Focus on running time as limited resource

At the end of the day, most interesting measure (?)

Today

Focus on memory usage

Second most fundamental resource (?)

Get new cplx classes and complete problems for them

Some similarities with time-bounded computation, but also some striking differences

DEF 1 A language $L \subseteq \{0,1\}^*$ is in $SPACE(s(n))$ if

$\exists$ constant c

Turing machine M

s.t. M decides L

M's heads never visit more than $c \cdot s(n)$ distinct locations on READ-WRITE TAPES EXCLUDING INPUT TAPE! for any input of length n.

$L \in NSPACE(s(n))$ if $\exists$ NDTM M deciding L s.t. at most $c \cdot s(n)$ read-write locations are visited for any input of length n and any non-det choices.

Important points

- Input stored on read-only input tape
  Doesn't count towards memory
  $\Rightarrow$ Possible to do computations in <u>sublinear space</u>

- Decision problem: Need not use output tape other than for answer yes/no (0 / 1).
  Focus on work tape(s)

- Look at only <u>space-constructible $s(n)$</u>
  $\exists$ TM that computes $s(|x|)$ in $O(s(|x|))$ space given input x. (Technical condition that we will ignore.)

- Space bounds of interest $\geq \log n$ — want TM to be able to remember positions on input tape.

Clearly $DTIME(s(n)) \subseteq SPACE(s(n))$
Can visit at most one tape position per time step
But space can be <u>reused</u>
Use space $S(n)$ to count from 0 to $2^{s(n)} - 1$
This is (almost) all that we know.

<u>THM 2</u>
$$DTIME(s(n)) \subseteq SPACE(s(n)) \subseteq NSPACE(s(n)) \subseteq DTIME(2^{O(s(n))}).$$

(Will be proven shortly.)

In fact, can do slightly better

THM 3 (Hopcroft, Paul, Valiant '77)

$$DTIME(s(n)) \subseteq SPACE(s(n)/\log s(n))$$

So space is strictly more powerful than time as resource. (Probably won't do anything close to proving Thm 3.)

One more point:

- What about termination in Def 1?
  Can require it. Not necessary, really.
  After $2^{O(s(n))}$ steps, workspaces has looked exactly the same, ~~read/write~~ heads have been exactly the same place, (ND)TM state has been exactly the same, etc at two time steps $t_1 < t_2$.
  So can ignore computation during interval $[t_1+1, t_2]$.
  If $\exists$ accepting computation, only $2^{O(s(n))}$ steps needed. Can equip any TM with "clock" that terminates after $2^{O(s(n))}$ steps. Only $O(s(n))$ space needed to count the time.

Back to proof of Thm 2 ...

DEF 4 Configuration graph of TM M

Configuration of M | consists of | at time t |

- program counter / state
- head positions
- contents of all tape positions that can possibly be visited (for same input length)

Configuration graph of M on input $x \in \{0,1\}^*$ | space $s(n)$ TM |

Directed graph $G_{M,x}$

Vertices: all possible configs C with input $= x$ and $c \cdot s(|x|)$ worktape cells

Edges: (C, C') if C' can be reached from C in one step acc to M's transition function.

Deterministic TM: out-degree 1

Nondeterministic TM out-degree 2

Assume M has "clean-up phase" erasing all worktapes before halting $\Rightarrow$ one unique accepting config C_{accept}.

M accepts $x \Leftrightarrow \exists$ path in $G_{M,x}$ from C_{start} to C_{accept}

CLAIM 5

1. Every vertex in $G_{M,x}$ can be described using $K \cdot s(n)$ bits ($K = O(1)$ depending on alphabet, # tapes, # states)
2. $G_{M,x}$ has at most $2^{O(s(n))}$ vertices
3. $\exists$ $O(s(n))$-size CNF formula $\varphi_{M,x}$ s.t. $\varphi_{M,x}(C, C') = 1$ iff (C, C') edge in $G_{M,x}$

Proof

1. Sort of by description in Def 4
2. Follows from 1.
3. Use Cook-Levin-style reasoning

Formula contains lots of local consistency checks
- tape contents are correct one time step later
- jump to correct state given read bits and previous state
- et cetera

$O(s(n))$ checks

Each check involves constant # bits = variables. $\boxed{\text{Ⓜ}}$

Proof of Thm 2

Only need to show $NSPACE(s(n)) \subseteq DTIME(2^{O(s(n))})$.
Construct $G_{M,x}$ in $2^{O(s(n))}$ time.
Do BFS to check if C_{accept} reachable from C_{start}. $\boxed{\text{Ⓜ}}$

DEF 6 Some cplx classes of particular interest

$$PSPACE = \bigcup_{c \in \mathbb{N}^+} SPACE(n^c)$$
$$NPSPACE = \bigcup_{c \in \mathbb{N}^+} NSPACE(n^c)$$

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

PROPOSITION 7 $NP \subseteq PSPACE$

Proof – Reduce to 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 $NP \neq L$?

EXAMPLE 9

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

PATH $\in NL$

Proof If there is a path, there is one of length $\leq n = |V(G)|$

Keep counter $[1 \ldots n]$ – $\log n$ 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 $O(\log n)$.] $\boxed{\text{H}}$

Is PATH in L? Excellent question

Would imply $L = NL$ (i.e., PATH is NL-complete; will be discussed later.

Interestingly [Reingold '05] proved that UNDIRECTED PATH is in L $\boxed{\text{H}}$ ☺

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

$$\mathrm{SPACE}(f(n)) \subsetneq \mathrm{SPACE}(g(n))$$

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

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

Proof Problem set 1.

Let's look at a more interesting problem

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

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

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

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

φ 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.

<u>Note</u> QBFs have determined truth value – either true or false.

<u>Example 13</u>

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

"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)

<u>THM 14</u> [Stockmeyer & Meyer '73]

The language

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

is PSPACE-complete

<u>Proof</u> ~~FBR~~ <u>TQBF $\in$ PSPACE</u> (sketch)

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

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

Inductive step

$\forall x_i \; \psi'$

set $x_1 = 0$, evaluate, save ← REUSING SPACE
set $x_1 = 1$, 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 $\Longleftrightarrow$ 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 $\leq 2^i$

24 : X

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

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

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

<u>Not good</u>: 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...

[JN]

24 : XI

A funny observation

Proof of Thm 14 established
that anything in PSPACE reduces to TQBF
via analysis of $G_{M,x}$.
<u>But</u> we never used out-degree 1 restriction...

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

So... TQBF is NPSPACE-hard.

<u>COROLLARY 15</u>

PSPACE = NPSPACE.

Can actually prove sth slightly more
precise

<u>THEOREM 16</u> (SAVITCH'S THEOREM '70)

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

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

<u>Proof sketch</u> 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 reusage $\Rightarrow$ space $O(s(n)^2)$ $\boxed{\text{Ez}}$

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 TIME

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