Lecture 21: Space Complexity  
(The Final Exam Frontier?)
For your favorite course on automata and complexity

Please complete the online subject evaluation for 6.045
Space Complexity
We measure space complexity by finding the largest tape index reached during the computation.
Let $M$ be a deterministic TM.

Definition: The space complexity of $M$ is the function $S : \mathbb{N} \rightarrow \mathbb{N}$, where $S(n)$ is the largest tape index reached by $M$ on any input of length $n$.

Definition: $\text{SPACE}(S(n)) = \{ L \mid L \text{ is decided by a Turing machine with } O(S(n)) \text{ space complexity} \}$
Theorem: 3SAT ∈ SPACE(n)

Proof Idea: Try all possible assignments to the (at most n) variables in a formula of length n. This can be done in O(n) space.

Theorem: NTIME(t(n)) is in SPACE(t(n))

Proof Idea: Try all possible computation paths of t(n) steps for an NTM on length-n input. This can be done in O(t(n)) space.
Space Hierarchy Theorem

Intuition: If you have more \textit{space} to work with, then you can solve strictly more problems!

Theorem: For functions \( s, S : N \rightarrow N \) where \( s(n)/S(n) \rightarrow 0 \)

\[
\text{SPACE}(s(n)) \nsubseteq \text{SPACE}(S(n))
\]

Idea: Diagonalization

Make a machine \( M \) that uses \( S(n) \) space and “does the opposite” of all \( O(s(n)) \) space machines on at least one input

So \( L(M) \) is in \( \text{SPACE}(S(n)) \) but not \( \text{SPACE}(s(n)) \)
PSPACE =$\bigcup_{k \in \mathbb{N}} $ \text{SPACE}(n^k)$

Since for every $k$, $\text{NTIME}(n^k)$ is in $\text{SPACE}(n^k)$, we have:

$P \subseteq \text{NP} \subseteq \text{PSPACE}$
The class PSPACE formalizes the set of problems solvable by computers with \textit{bounded memory}.

**Fundamental (Unanswered) Question:** How does time relate to space, in computing?

SPACE($n^2$) problems could potentially take much longer than $n^2$ time to solve!

\textit{Intuition: You can always re-use space, but how can you re-use time?}

Is $P = \text{PSPACE}$?
Time Complexity of $\text{SPACE}[S(n)]$

Let $M$ be a halting TM that on input $x$, uses $S$ space

How many time steps can $M(x)$ possibly take? Is there an upper bound?

The number of time steps is at most the total number of possible *configurations* of $M$!

*(If a configuration repeats, the machine is looping.)*

A configuration of $M$ on $x$ specifies a head position, a state, and $S$ cells of tape content. So the total number of configurations is at most:

$$S \cdot |Q| \cdot |\Gamma|^S \leq 2^{O(S)}$$
Theorem:
Space $S(n)$ computations can be decided in $2^{O(S(n))}$ time

$$\text{SPACE}(s(n)) \subseteq \bigcup_{c \in \mathbb{N}} \text{TIME}(2^c \cdot s(n))$$

Idea: After $2^{O(s(n))}$ time steps, a $s(n)$-space bounded computation must have repeated a configuration, so then it will never halt.
PSPACE = \bigcup_{k \in \mathbb{N}} \text{SPACE}(n^k)

EXPTIME = \bigcup_{k \in \mathbb{N}} \text{TIME}(2^{n^k})

PSPACE \subseteq \text{EXPTIME}
P \subseteq NP \subseteq \text{PSPACE}

Is NP^{NP} \subseteq \text{PSPACE}?

YES

And coNP^{NP} \subseteq \text{PSPACE}
Example: MIN-FORMULA is in PSPACE

MIN-FORMULA = \{ \phi \mid \phi \text{ is minimal} \}

Recall the coNP^{NP} algorithm for MIN-FORMULA:

Given a formula \( \phi \),

Try all formulas \( \psi \) such that \( \psi \) is smaller than \( \phi \).
If \((\phi, \psi) \in \text{NEQUIV})\) then accept else reject

Can store a formula \( \psi \) in space \( O(|\phi|) \)
Can check \((\phi, \psi) \in \text{NEQUIV} \) by trying all assignments to the variables of \( \phi \) and \( \psi \)
Can store a variable assignment in space \( O(|\phi|) \)
Evaluating \( \psi \) or \( \phi \) on an assignment is \( O(|\phi|) \) space
\( P \subseteq NP \subseteq PSPACE \subseteq EXPTIME \)

Theorem: \( P \neq EXPTIME \)

Why? The Time Hierarchy Theorem!

\( \text{TIME}(2^n) \not\subseteq P \)

Therefore \( P \neq EXPTIME \)

Corollary: At least one of the following is true: 
\( P \neq NP, \quad NP \neq PSPACE, \quad \text{or} \quad PSPACE \neq EXPTIME \)

Proving any one of them would be major!
A Closer Look at PSPACE and Nondeterminism
Definition: \( \text{SPACE}(s(n)) = \{ L \mid L \text{ is decided by a Turing machine with } O(s(n)) \text{ space complexity} \} \)

Definition: \( \text{NSPACE}(s(n)) = \{ L \mid L \text{ is decided by a } \text{non-deterministic Turing Machine with } O(s(n)) \text{ space complexity} \} \)
Recall:
Space $S(n)$ computations can be simulated in at most $2^{O(S(n))}$ time steps

$$\text{SPACE}(s(n)) \subseteq \bigcup_{c \in \mathbb{N}} \text{TIME}(2^c \cdot s(n))$$

Idea: After $2^{O(s(n))}$ time steps, a $s(n)$-space bounded computation must have repeated a configuration, so then it will never halt.
Theorem:
NSPACE $S(n)$ computations can also be simulated in at most $2^{O(S(n))}$ time steps

$$\text{NSPACE}(s(n)) \subseteq \bigcup_{c \in \mathbb{N}} \text{TIME}(2^c \cdot s(n))$$

Key Idea: Think of the problem of simulating NSPACE$(s(n))$ as a problem on graphs.
Def: The configuration graph of M on x has nodes $C$ for every configuration $C$ of M on x, and edges $(C, C')$ if and only if $C$ yields $C'$

$G_{M,x}$

M has space complexity $S(n)$
$\Rightarrow G_{M,x}$ has $2^{c \cdot S(|x|)}$ nodes

M is deterministic
$\Rightarrow$ every node has outdegree $\leq 1$

M is nondeterministic
$\Rightarrow$ some nodes may have outdegree $> 1$

To simulate a non-deterministic M in $2^{O(S(|x|))}$ time: do BFS in $G_{M,x}$ from the initial configuration!
PSPACE = $\bigcup_{k \in \mathbb{N}} \text{SPACE}(n^k)$

NPSPACE = $\bigcup_{k \in \mathbb{N}} \text{NSPACE}(n^k)$
SPACE versus NSPACE

Is NTIME(n) ⊆ TIME(n^2)?

Is NTIME(n) ⊆ TIME(n^k) for some k > 1?

Nobody knows!

If the answer is yes, then P = NP...
What about the space-bounded setting?

Does NSPACE(s(n)) ⊆ SPACE(s(n)^k) for some k?