Intro to Theory of Computation

LECTURE 25

Last time
• Cook-Levin Theorem
• Examples of NP-complete languages

Today
• Space complexity
• The class PSPACE

Sofya Raskhodnikova
1. $3\text{SAT} \leq_p A_{TM}$
2. $A_{TM}$ is NP-complete

A. (1) and (2) are both true
B. (1) and (2) are both false
C. (1) is true and (2) is false
D. (2) is true and (1) is false
E. At least one of (1) and (2) is an open question
Space analysis

If $M$ is a TM and $f: \mathbb{N} \to \mathbb{N}$ then

“$M$ runs in space $f(n)$” means

for every input $w \in \Sigma^*$ of length $n$,

$M$ on $w$ uses at most $f(n)$ tape cells.

- If $M$ is a nondeterministic TM that halts on all inputs then
  $f(n)$ is the maximum number of cells $M$ uses on any
  input of length $n$. 

Space complexity classes

SPACE($f(n)$) is a class of languages. $A \in \text{SPACE}(f(n))$ means that some 1-tape TM $M$ that runs in space $O(f(n))$ decides $A$. 
Prove: SAT \in\text{SPACE}(n)

M = ```
On input \langle \phi \rangle, where \phi is a Boolean formula, with variables \(x_1, \ldots, x_\ell\):

1. For each truth assignment to \(x_1, \ldots, x_\ell\)
2. Evaluate \phi on that truth assignment.
3. Accept if \phi ever evaluates to 1. O.w. reject.”
```  

- If \(n\) is the input length, M uses space \(O(n)\).
Space complexity classes

\[ \text{NSPACE}(f(n)) \] is a class of languages.  

\( A \in \text{NSPACE}(f(n)) \) means that some 1-tape nondeterministic TM \( M \) that runs in space \( O(f(n)) \) decides \( A \).
Prove: \( \text{ALL}_{NFA} \in \text{NSPACE}(n) \)

- \( \text{ALL}_{NFA} = \{ \langle M \rangle | M \text{ is an NFA and } L(M) = \Sigma^* \} \)

\(N = \) ``On input \( \langle M \rangle \), where \( M \) is an NFA:

1. Place marker on the start state of \( M \).
2. Repeat \( \square \) times where \( q \) is the # of states of \( M \):
   3. Nondeterministically select \( a \in \Sigma \).
   4. Adjust the markers to simulate \( M \) reading \( a \).
5. Accept if at any point none of the markers are on an accept state. O.w. reject."

- If \( n \) is the input length, \( N \) is an NTM that uses space \( O(n) \).
Savitch’s theorem

**Theorem.** Let $f(n)$ be a function, where $f(n) \geq n$. 
NSPACE($f(n)$) $\subseteq$ SPACE($f^2(n)$).
Savitch’s theorem

**Theorem.** Let \( f(n) \) be a function, where \( f(n) \geq n \). \( \text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n)) \).

**Proof:**

- Let \( N \) be an NTM deciding a language \( A \) in \( f(n) \) space.
- We give a deterministic TM \( M \) deciding \( A \).
- More general problem:
  - Given configurations \( c_1, c_2 \) of \( N \) and integer \( t \), decide whether \( N \) can go from \( c_1 \) to \( c_2 \) in \( \leq t \) steps on some nondeterministic path.
  - Procedure \( \text{CANYIELD}(c_1, c_2, t) \)
Savitch’s theorem

**Theorem.** Let $f(n)$ be a function, where $f(n) \geq n$. $\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n))$.

**Proof:** (the rest of the proof on the board)

**CANYIELD** = `On input \(\langle c_1, c_2, t \rangle\):

1. If \(t = 1\), accept if \(c_1 = c_2\) or \(c_1\) yields \(c_2\) in one transition. O.w. reject.
2. If \(t > 1\), then \(\forall\) configs \(c_{\text{mid}}\) of \(N\) with \(\leq f(n)\) cells:
   3. Run CANYIELD(\(\langle c_1, c_{\text{mid}}, t/2 \rangle\)).
   4. Run CANYIELD(\(\langle c_{\text{mid}}, c_2, t/2 \rangle\)).
5. If both runs accept, accept.
6. Reject.”
The class PSPACE

**PSPACE** is the class of languages decidable in polynomial space on a *deterministic* TM:

\[
PSPACE = \bigcup_{k} \text{SPACE}(n^k).
\]

- NPSPACE – the same, but for NTMs.
- By Savitch’s Theorem,

\[
PSPACE = \text{NPSPACE}
\]
Relationships between classes

1. \( P \subseteq \text{NP} \subseteq \text{PSPACE} \subseteq \text{EXPTIME} \)
   - **Recall**: a TM that runs in space \( f(n) \) has \( \leq f(n)2^{O(f(n))} \) configurations

2. \( P \neq \text{EXPTIME} \)
   - Which containments in (1) are proper?
   - **Unknown!**
A language $B$ is **PSPACE-complete** if

1. $B \in \text{PSPACE}$
2. $B$ is **PSPACE-hard**, i.e., every language in PSPACE is poly-time reducible to $B$. 
The TQBF problem

- **Boolean variables**: variables that can take on values T/F (or 1/0)
- **Boolean operations**: ∨, ∧, and ¬
- **Boolean formula**: expression with Boolean variables and ops
- **Quantified Boolean formula**: Boolean formula with quantifiers (∀, ∃)
- **Fully Quantified Boolean formula**: all variables have quantifiers (∀, ∃)
  We only consider the form where all quantifiers appear in the beginning.

\[ \exists y \forall x [ (x \lor y) \land (\overline{x} \lor \overline{y}) ] \quad \text{True} \]
\[ \forall x \exists y [ (x \lor y) \land (\overline{x} \lor \overline{y}) ] \quad \text{False} \]

- Each fully quantified Boolean formula is either true or false.
- The order of quantifiers matters!

**TQBF** = \{⟨φ⟩ | φ is a true fully quantified Boolean formula\}
TQBF is PSPACE-complete

1. TQBF is in PSPACE
2. TQBF is PSPACE-hard
Prove: \( \text{TQBF} \in \text{PSPACE} \)

\[ T = \text{``On input } \langle \phi \rangle, \text{ where } \phi \text{ is a fully quantified Boolean formula:} \]

1. If \( \phi \) has no quantifiers, it has only constants (and no variables). Evaluate \( \phi \). If true, accept; o.w., reject.

2. If \( \phi \) is of the form \( \exists x \psi \), recursively call \( T \) on \( \psi \) with \( x = 0 \) and then on \( \psi \) with \( x = 1 \). If either call accepts, accept; o.w., reject.

3. If \( \phi \) is of the form \( \forall x \psi \), recursively call \( T \) on \( \psi \) with \( x = 0 \) and then on \( \psi \) with \( x = 1 \). If both calls accept, accept; o.w., reject.”

• If \( n \) is the input length, \( T \) uses space \( O(n) \).
If TQBF is in P then it implies that

A. $P = NP$
B. $P = \text{PSPACE}$
C. $P = \text{EXPTIME}$
D. (A) and (B) are true
E. (A), (B), (C) are true