- PSPACE Completeness (Sipser 8.3)
- L and NL (Sipser 8.4, 8.5)

Some material from slides by M. Sipser
Announcements

- UPE Announcement

- No class on Thursday, April 1.
Exam 2

- Monday, March 29, 2021
  - Will be given through Gradescope.
  - Available 12pm – 8pm NY Time. You will have 2 hours.
  - Exam is open book and open notes. No collaboration. No posting questions or searching for solutions online.
  - Exam is not cumulative, but will use concepts from pre-Exam 1 (e.g., DFAs, NFAs, TMs, Regular Expressions, Turing recognizability, decidability)
    - Similar to the way we have used these topics in class and on homework problems.
Chapter 4: Undecidability (Section 4.2)

Chapter 5: Reducibility, Mapping Reducibility
  - No questions on the Post Correspondence Problem

Chapter 6: The Recursion Theorem (Section 6.1)
  - Both the proof and applications of the theorem

Chapter 7: The classes TIME(t(n)), NTIME(t(n)), P, NP, polynomial time reductions, NP-completeness, the Cook-Levin Theorem
  - Also refresh your memory on problems: SAT, 3SAT, CLIQUE, PATH, HAMPATH

Chapter 8: Savitch’s Theorem, PSPACE, NPSPACE, PSPACE-completeness
  - No questions on Sections 8.4 – 8.6.
Review – Space Complexity Classes

- **Deterministic TM**
  - The **space complexity** of $M$ is a function $f: \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$, where $f(n)$ is the maximum number of tape cells $M$ accesses on any input of length $n$.
  - $SPACE(f(n))$ is the set of languages that are decidable in $O(f(n))$ space.
  - $PSPACE = \bigcup_k SPACE(n^k)$

- **Nondeterministic TM**
  - The **space complexity** of $N$ is a function $f: \mathbb{N} \to \mathbb{N}$ with $f(n) \geq n$, where $f(n)$ is the maximum number of tape cells $N$ accesses on a **single computation path** on any input of length $n$.
  - $NSPACE(f(n))$ is the set of languages that are decidable in $O(f(n))$ space.
  - $NPSPACE = \bigcup_k NSPACE(n^k)$
Review 1 – Relationship Between Space and Time Complexity

- **Theorem:**
  1) $TIME(t(n)) \subseteq SPACE(t(n))$
  2) $SPACE(t(n)) \subseteq TIME(2^{O(t(n)}) = \bigcup_c TIME(c^{t(n)})$
Review 2 – Relationship Between Space and Time Complexity

- Theorem: $P \subseteq NP \subseteq PSPACE$
Review– Savitch’s Theorem

- **Theorem:** For any function \( f(n): \mathbb{N} \rightarrow \mathbb{R}^+ \) where \( f(n) \geq n \), \( NSPACE(f(n)) \subseteq SPACE(f^2(n)) \).

- Proof idea: Convert NTM \( N \) to equivalent TM \( M \), only squaring the space used.

- \( M \): On input \((c_i, c_j, t)\)
  1. If \( t = 1 \), check if \( c_j \) can be reached from \( c_i \) using \( N \)'s transition function in 0 or 1 steps. Accept if yes; otherwise, reject.
  2. If \( t > 1 \), repeat for all configurations \( c_m \) that use \( f(n) \) space.
  3. Recursively test \((c_i, c_m, t/2)\) and \((c_m, c_j, t/2)\)
  4. If both accept, then accept. If not, continue.
  5. Reject if haven’t yet accepted.

- Test if \( N \) accepts \( w \) by testing \( M \) on input \((c_{\text{start}}, c_{\text{accept}}, t)\) where \( t = \) number of configurations.
As a result of Savitch’s theorem: $PSPACE = NPSPACE$

What we’ve learned so far: $P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME$
A language $B$ is PSPACE-complete if:

1. $B \in PSPACE$
2. For all $A \in PSPACE$, $A \leq_p B$

We will show that TQBF is PSPACE-Complete
TQBF is PSPACE-Complete

- Recall: $TQBF = \{ \langle \phi \rangle \mid \phi \text{ is a QBF that is } \text{TRUE} \}$

- Last lecture – we showed TQBF is in PSPACE.
  - Gave a recursive algorithm that “peels off” one quantifier at each level of recursion.
  - Each level uses $O(1)$ space; the recursion depth is $\leq n$.

- Now – we will give a polynomial time reduction from arbitrary language $A \in PSPACE$ to $TQBF$. 
Constructing \( \phi \) - First Attempt

- Use approach like in proof of Cook-Levin theorem.
- Construct formula \( \phi \) that matches an accepting tableau.

\[
\begin{array}{|c|c|c|c|}
\hline
q_0 & w_1 & w_2 & \cdots w_n \\
\hline
a & q_7 & w_2 & \cdots \\
\hline
\cdots & q_{\text{accept}} & \cdots \\
\hline
\end{array}
\]
Constructing $\phi$ - Second Attempt

- Use an approach like in Proof of Savitch’s Theorem – divide and conquer
  - Encode contents of configuration cells (from tableau) as Boolean expression like in Cook-Levin theorem
  - Recursively construct $\phi_{c_{\text{start}},c_{\text{accept}},t}$ like in Savitch’s theorem
Constructing $\phi$ - Third Attempt

- Use an approach like in proof of Savitch’s Theorem – divide and conquer
  - Encode contents of configuration cells (from tableau) as Boolean expression like in Cook-Levin theorem
  - Recursively construct $\phi_{c_{\text{start}},c_{\text{accept}},t}$ like in Savitch’s theorem
Summary – So Far

PSPACE = NPSPACE

NP-complete

PSPACE-complete

NP

P
The Classes L and NL
Sublinear Space Complexity

- So far, we have studied space complexity bounds that are at least linear: $f(n) \geq n$.

- Are there problems that can be solved in sublinear space, $f(n) < n$?

- To study sublinear space algorithms, we need a different model.
  - Two-tape TM with read-only input tape and read/write work tape.

read-only input tape does not count towards space used

count cells used here

read/write work tape
We focus on algorithms with space complexity in $O(\log n)$

$L$ is the class of languages that are decidable in logarithmic space on a deterministic TM:

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

$N$ is the class of languages that are decidable in logarithmic space on a nondeterministic TM:

$$NL = NSPACE(\log n)$$
Example in Class L

- $A = \{0^k 1^k \mid k \geq 0 \}$
Example in Class NL

- \( PATH = \{ \langle G, s, t \rangle \mid G \text{ is a directed graph that has a directed path from } s \text{ to } t \} \)
Relationship to Other Complexity Classes

- **Theorem**: $NL \subseteq P$

- **Theorem**: $NL \subseteq \text{SPACE}(\log^2 n)$