Space Complexity

CS60001: Foundations of Computing Science

Pallab Dasgupta
Professor, Dept. of Computer Sc. & Engg.,
Indian Institute of Technology Kharagpur

Introduction

- **Definition**
  - Let *M* be a deterministic Turing machine that halts on all inputs. The *space complexity* of *M* is the function $f: \mathcal{N} \rightarrow \mathcal{N}$, where $f(n)$ is the maximum number of tape cells that *M* scans on any input of length *n*. If the space complexity of *M* is $f(n)$, we also say that *M* runs in space $f(n)$

  - If *M* is a non-deterministic Turing machine wherein all branches halt on all inputs, we define its space complexity $f(n)$ to be the maximum number of tape cells that *M* scans on any branch of its computation for any input of length *n*

Space Complexity Classes

- **Definition:** Let $f: \mathcal{N} \rightarrow \mathcal{R}^+$ be a function. The space complexity classes, *SPACE(f(n))* and *NSPACE(f(n))*, are defined as follows:
  - SPACE(f(n)) = {L | L is a language decided by O(f(n)) space deterministic Turing machine}
  - NSPACE(f(n)) = {L | L is a language decided by an O(f(n)) space non-deterministic Turing machine}
- **Examples**
  - *SAT* can be solved with a linear space algorithm [Space complexity = O(n)]
  - Testing whether a non-deterministic finite automaton accepts all strings, i.e. $ALL_{NFA} = \{<\mathcal{A}> \mid \mathcal{A} \text{ is a NFA and } L(A) = \Sigma^*\}$
    - Non-deterministic space complexity = O(n)
- **SAVITCH'S Theorem**
  - For any function $f: \mathcal{N} \rightarrow \mathcal{R}^+$, where $f(n) \geq n$,

$$\text{NSPACE}(f(N)) \subseteq \text{SPACE}(f^2(n))$$

The Class PSPACE

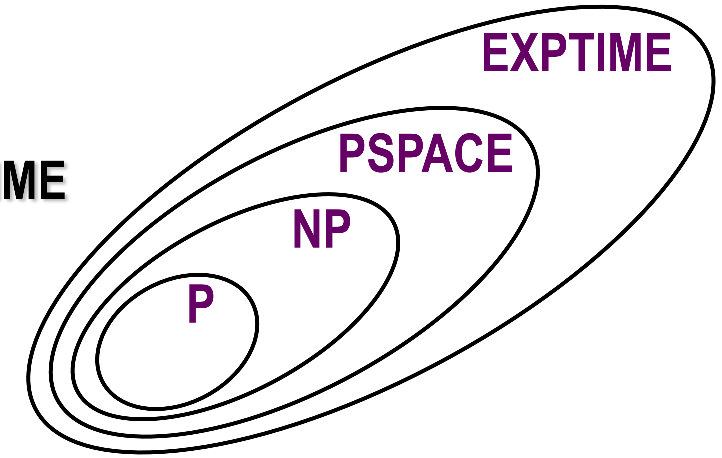
- **Definition**
  - *PSPACE* is the class of languages that are decidable in polynomial space on a deterministic Turing machine. In other words,

$$\text{PSPACE} = \bigcup_k \text{SPACE}(n^k)$$

  - *NPSPACE* is the class of languages that are decidable in polynomial space on a non-deterministic Turing machine. In other words,

$$\text{NPSPACE} = \bigcup_k \text{NSPACE}(n^k)$$

- **Relationship among the Complexity Classes**
  - $\text{P} \subseteq \text{NP} \subseteq \text{PSPACE} = \text{NPSPACE} \subseteq \text{EXPTIME}$



PSPACE-Completeness

- **Definition**
  - A language $\mathcal{B}$ is *PSPACE-complete* if it satisfies two conditions:
    - $\mathcal{B}$ is in PSPACE, and
    - Every $\mathcal{A}$ in PSPACE is polynomial time reducible to $\mathcal{B}$
  - If $\mathcal{B}$ merely satisfies condition-2, we say that it is *PSPACE-hard*

- **Examples of PSPACE-complete Problems**
  - TQBF = {<Φ> | Φ is a true fully quantified Boolean formula}
  - FORMULA-GAME = {<Φ> | Player *E* has a winning strategy in the formula game associated with *Φ*}
  - GENERALIZED-GEOGRAPHY =
    {<G, b> | Player *I* has a winning strategy for the generalized geography game played on the graph G starting at node *b*}

The Classes L and NL

- **Definitions**
  - *L* is the class of languages that are decidable in logarithmic space on a deterministic Turing machine. In other words, L = SPACE(log n)
  - *NL* is the class of languages that are decidable in logarithmic space on a non-deterministic Turing machine. In other words, NL = NSPACE(log n)
- **Examples**
  - The language $\mathcal{A} = \{0^k 1^k \mid k \geq 0\}$ is a member of *L*
  - The language PATH = {<G, s, t> | G is a directed graph that has a directed path from s to t} is a member of *NL*
- **Definition**
  - If *M* is a Turing machine that has a separate read-only input tape and *w* is an input, a configuration of *M* on *w* is a setting of the state, the work tape, and the position of the two tape heads. The input *w* is not a part of the configuration of *M* on *w*

Log-Space Reducibility

- **Definitions**
  - A *log space transducer* is a Turing machine with a read-only input tape, a write-only output tape, and a read/write work tape. The work tape may contain $O(\log n)$ symbols.
  - A log space transducer M computes a function $f: \Sigma^* \rightarrow \Sigma^*$, where $f(w)$ is the string remaining on the output tape after M halts when it is started with w on its input tape. We call f a *log space computable function*.
  - Language $\mathcal{A}$ is *log space reducible* language $\mathcal{B}$, written $\mathcal{A} \leq_L \mathcal{B}$, if $\mathcal{A}$ is mapping reducible to $\mathcal{B}$ by means of a log space computable function f

NL-Completeness

- **Definition**
  - A language $\mathcal{B}$ is *NL-complete* if
    - $\mathcal{B} \in$ NL, and
    - Every $\mathcal{A}$ in NL is log space reducible to $\mathcal{B}$
- **Theorem**
  - If $\mathcal{A} \leq_L \mathcal{B}$ and $\mathcal{B} \in$ L, then $\mathcal{A} \in$ L
  - Corollary: If any NL-complete language is in *L*, then *L* = *NL*
- **Example of NL-complete Problems**
  - PATH = {<G, s, t> | G is a directed graph that has a directed path from s to t}
  - Corollary: NL $\subseteq$ P
- **Theorem**
  - NL = coNL