Complexity Classes

Chapter 8 (Abbreviated)

Overview

- Problems are classified according to the resources they use on computing machines (serial or parallel)
- Serial models are RAM and Turing machines
- Parallel models are the circuit and PRAM
- Terms of discussion are first defined
  - Tasks that are well-defined
  - Resource measures
  - Machine models
- Serial complexity classes are discussed
  - \( \mathbf{P} \)-complete problems
  - \( \mathbf{NP} \)-complete problems
  - \( \mathbf{PSPACE} \)-complete problems
Definitions and Problems

Definition 8.2.1 Let $\Sigma$ be an arbitrary finite alphabet. A decision problem $P$ is defined by a set of instances $I \subseteq \Sigma^*$ of the problem and a condition $\phi_P : I \rightarrow \mathbb{B}$ that has value 1 on "Yes" instances and 0 on "No" instances. Then $I_{\text{yes}} = \{w \in I \mid \phi_P(w) = 1\}$ are the "Yes" instances. The "No" instances are $I_{\text{no}} = I - I_{\text{yes}}$.

The complement of a decision problem $P$, denoted $\text{co}P$, is the decision problem in which the "Yes" instances of $\text{co}P$ are the "No" instances of $P$ and vice versa.

The "Yes" instances of a decision problem are encoded as binary strings by an encoding function $\sigma : \Sigma^* \rightarrow \mathbb{B}^*$ that assigns to each $w \in I$ a string $\sigma(w) \in \mathbb{B}^*$.

With respect to $\sigma$, the language $L(P)$ associated with a decision problem $P$ is the set $L(P) = \{\sigma(w) \mid w \in I_{\text{yes}}\}$. With respect to $\sigma$, the language $L(\text{co}P)$ associated with $\text{co}P$ is the set $L(\text{co}P) = \{\sigma(w) \mid w \in I_{\text{no}}\}$.

The complement of a language $L$, denoted $L^c$, is $B^* - L$; that is, $L$ consists of the strings that are not in $L$.

A decision problem can be generalized to a problem $P$ characterized by a function $f : \mathbb{B}^* \rightarrow \mathbb{B}^*$ described by a set of ordered pairs $(x, f(x))$, where each string $x \in \mathbb{B}^*$ appears once as the left-hand side of a pair. Thus, a language is defined by problems $f : \mathbb{B}^* \rightarrow \mathbb{B}$ and consists of the strings on which $f$ has value 1.

Resource Bounds

- “Feasible” problems are defined currently as those problems which can be solved with a deterministic TM in polynomial time – known as the serial computation thesis.
- Problems can be classified according to their use and requirement of resources, $r(n)$.
- Resource bounds expressed in terms of these functions:
  - Logarithmic function: $r(n) = O(\log n)$
  - Poly-logarithmic function: $r(n) = \log^{O(1)} n$
  - Linear function: $r(n) = O(n)$
  - Polynomial function: $r(n) = n^{O(1)}$
  - Exponential function: $r(n) = 2^{n^{O(1)}}$
Serial Computational Models

- **Random-Access Machine**: this flavor allows for words that have potentially unbounded length

Serial Computational Models (cont.)

- **Turing Machine**: will use deterministic (DTM) and non-deterministic (NDTM) versions, as well as multi-tape flavors of each
Classification of Decision Problems

**Definition 8.5.1** Let \( r(n) : \mathbb{N} \to \mathbb{N} \) be a proper resource function. Then \( \text{TIME}(r(n)) \) and \( \text{SPACE}(r(n)) \) are the \textit{time and space Turing complexity classes} containing languages that can be recognized by DTM's that halt on all inputs in time and space \( r(n) \), respectively, where \( n \) is the length of an input. \( \text{NTIME}(r(n)) \) and \( \text{NSPACE}(r(n)) \) are the \textit{nondeterministic time and space Turing complexity classes}, respectively, defined for NDTMs instead of DTM's. The union of complexity classes is also a complexity class.

**Definition 8.5.2** The classes \( P \) and \( NP \) are sets of decision problems solvable in polynomial time on DTM's and NDTM's, respectively; that is, they are defined as follows:

\[
P = \bigcup_{k \geq 0} \text{TIME}(n^k)
\]

\[
NP = \bigcup_{k \geq 0} \text{NTIME}(n^k)
\]

- Problems in \( P \) are currently defined as “feasible” problems

Classification of Decision Problems (cont.)

**Definition 8.5.3** The classes \( \text{EXPTIME} \) and \( \text{NEXPTIME} \) consist of those decision problems solvable in deterministic and nondeterministic exponential time, respectively, on a Turing machine. That is,

\[
\text{EXPTIME} = \bigcup_{k \geq 0} \text{TIME}(k^n)
\]

\[
\text{NEXPTIME} = \bigcup_{k \geq 0} \text{NTIME}(k^n)
\]

**Theorem 8.5.4** The following complexity class containments hold:

\[
P \subseteq NP \subseteq \text{EXPTIME} \subseteq \text{NEXPTIME}
\]

However, \( P \subseteq \text{EXPTIME} \), that is, \( P \) is strictly contained in \( \text{EXPTIME} \).

- Is this last result evidence that \( P \neq \text{NP} \)?
Reductions

- Generalize the notion of reductions to include bounds on resources needed and preservation of complexity class.

**Definition 8.7.1** If $L_1$ and $L_2$ are languages, a transformation $h$ from $L_1$ to $L_2$ is a DTM-computable function $h : B^* \to B^*$ such that $x \in L_1$ if and only if $h(x) \in L_2$. A resource-bounded transformation is a transformation that is computed under a resource bound such as deterministic logarithmic space or polynomial time.

- Define new notation and terminology to include these notions:

**Definition 8.7.2** For decision problems $P_1$ and $P_2$, the notation $P_1 \leq_R P_2$ means that $P_1$ can be transformed to $P_2$ by a transformation in the class $R$.

**Definition 8.7.3** Let $C$ be a complexity class, $R$ a class of resource-bounded transformations, and $P_1$ and $P_2$ decision problems. A set of transformations $R$ is compatible with $C$ if $P_1 \leq_R P_2$ and $P_2 \in C$, then $P_1 \in C$.

Hard and Complete Problems

- Define the notions of “hard” and “complete” problems for different complexity classes.

**Definition 8.8.2** Let $R$ be a class of reductions, let $C$ be a complexity class, and let $R$ be compatible with $C$. A problem $Q$ is hard for $C$ under $R$-reductions if for every problem $P \in C$, $P \leq_R Q$. A problem $Q$ is complete for $C$ under $R$-reductions if it is hard for $C$ under $R$-reductions and is a member of $C$.

**Definition 8.8.3** Problems in $P$ that are hard for $P$ under log-space reductions are called $P$-complete. Problems in $NP$ that are hard for $NP$ under polynomial-time reductions are called $NP$-complete. Problems in $PSPACE$ that are hard for $PSPACE$ under polynomial-time reductions are called $PSPACE$-complete.

- The following result follows naturally from our definitions:

**Theorem 8.8.2** If a $P$-complete problem can be solved in log-space, then all problems in $P$ can be solved in log-space. If an $NP$-complete problem is in $P$, then $P = NP$. If a $PSPACE$-complete problem is in $P$, then $P = PSPACE$. 
P-Complete Problems

- These problems are P-Complete (there are hundreds of others):

  **Circuit Value**
  *Instance:* A circuit description with fixed values for its input variables and a designated output gate.
  *Answer:* "Yes" if the output of the circuit has value 1.

  **Monotone Circuit Value**
  *Instance:* A description for a monotone circuit with fixed values for its input variables and a designated output gate.
  *Answer:* "Yes" if the output of the circuit has value 1.

  **Linear Inequalities**
  *Instance:* An integer-valued $m \times n$ matrix $A$ and column $m$-vector $b$.
  *Answer:* "Yes" if there is a rational column $n$-vector $x > 0$ (all components are non-negative and at least one is non-zero) such that $Ax \leq b$.

NP-Complete Problems

- These problems are NP-Complete (there are literally thousands of others):

  **Circuit SAT**
  *Instance:* A circuit description with $n$ input variables $\{x_1, x_2, \ldots, x_n\}$ for some integer $n$ and a designated output gate.
  *Answer:* "Yes" if there is an assignment of values to the variables such that the output of the circuit has value 1.

  **Independent Set**
  *Instance:* A graph $G = (V, E)$ and an integer $k$.
  *Answer:* "Yes" if there is a set of $k$ vertices of $G$ such that there is no edge in $E$ between them.