**What Is Computation?**

Computation is a systematic way of obtaining an answer to a problem. The systematic nature of computation allows the use of computing devices for actual computation.

- “Is this model ‘different’ from that model?”
- “What problems can be solved under this model?”

**Problem Classes**

“Computation model” + “Concept of Solving Problems” = “Class of Problems”

- “Is this model and that model different?” → “Is class A equal to class B?”
- “Can any problem be solved under this model?” → “Exactly what is class A?”

**Computation Resources**

Distinction between resource-bounded computation and resource-unbounded computation

How does bounding resources affect the power of computation?
Alphabet, Strings, Languages, etc.

See page 16 of the textbook

Alphabet, Strings, Languages, etc.

- An alphabet is any finite set, whose members are called symbols.
- A string (or word) over an alphabet is a sequence of symbols from the alphabet written one after another.
- The length of a word \( w \), denoted by \( |w| \), is the number of symbols in it.
- The empty string, denoted by \( \epsilon \), is the string with no symbols in it.

Alphabet, Strings, Languages, etc. (cont’d)

- A string \( z \) is a substring of \( w \) if \( z \) appears consecutively within \( w \).
- The concatenation of strings \( x \) and \( y \) is the string constructed by appending \( y \) after \( x \).
- A language is a collection of strings.
- The complement of a language is the collection of all non-members.
- A class is a collection of languages.

Class Overview: Resource Unbounded

The Universe of Languages
- Recursively Enumerable
  - Recursive
    - Context-Free
      - Regular

Class Overview: Resource Bounded

RECURSIVE
- PSPACE
  - NP
    - P
      - NL
        - L
Boolean Logic (cont’d)

A **predicate** is a **function** whose **range** is TRUE, FALSE. A **relation** is a predicate whose number of arguments is fixed to a constant.

Properties of binary relation $R$ over domain $D$:

- **reflexive** for all $x \in D$, $xRx$
- **symmetric** for all $x, y \in D$, $xRy \leftrightarrow yRx$
- **transitive** for all $x, y, z \in D$, $xRy \land yRz \rightarrow xRz$

An equivalence relation is a binary relation that is reflexive, symmetric, and transitive.

### Alphabet, Strings, Languages, etc. (cont’d)

**Example** Let $\Sigma = \{0, 1\}$ be an alphabet. The symbols of $\Sigma$ are 0 and 1.

Let $z = 00111010$. Then $z$ is a string over $\Sigma$. 1111 is a substring of $z$ while 11111 is not. The concatenation of $a = 000$ and $b = 111$ is 000111.

The set of all strings over $\Sigma$ with the same number of 0s and 1s is a language. The collection of all languages over $\Sigma$ is a class.

Boolean Logic

A **Boolean variable** takes on one of 0 (FALSE) and 1 (TRUE). The **negation** of $x$, denoted by $\neg x$ or $\overline{x}$, is $1 - x$.

We will be using six binary Boolean operators:

<table>
<thead>
<tr>
<th>$(x, y)$</th>
<th>(0, 0)</th>
<th>(0, 1)</th>
<th>(1, 0)</th>
<th>(1, 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\land$</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$\lor$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>$\rightarrow$</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$\leftrightarrow$</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>$\oplus$</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>