Computability and Complexity Theory

Computability and complexity

- **Computability theory**
  - What is an algorithm?
  - What problems can be solved on a computer?
  - What is a computable function?
  - Solvable vs. unsolvable problems (decidability)

- **Complexity theory**
  - How much time and memory is needed to solve a problem?
  - Tractable vs. intractable problems

**What is a computable function?**

- Non-trivial question $\Rightarrow$ various formalizations, e.g.
  - General recursive functions $\quad$ Gõdel/Herbrand/Kleene 1936
  - $\lambda$-calculus $\quad$ Church 1936
  - $\mu$-recursive functions $\quad$ Gõdel/Kleene 1936
  - Turing machines $\quad$ Turing 1936
  - Post systems $\quad$ Post 1943
  - Markov algorithms $\quad$ Markov 1951
  - Unlimited register machines $\quad$ Shepherdson-Sturgis 1963
  ...

- All these approaches have turned out to be equivalent.

**Church-Turing thesis**

The class of intuitively computable functions is equal to the class of Turing computable functions.

**Finite automata**

*Finite automaton* $M = (Q, \Sigma, \delta, q_0, F)$ with

- $Q$ finite set of states
- $\Sigma$ finite input alphabet
- $\delta : Q \times \Sigma \to Q$ transition function
- $q_0 \in Q$ initial state
- $F \subseteq Q$ set of final states
Example

\[ M_0 = (Q, \Sigma, \delta, q_0, F) \] with

- \( Q = \{ q_0, q_1 \} \), \( \Sigma = \{ a, b \} \), \( F = \{ q_0 \} \)
- \( \delta(q_0, a) = q_0 \), \( \delta(q_0, b) = q_1 \), \( \delta(q_1, a) = q_1 \), \( \delta(q_1, b) = q_0 \)

### Recognizing languages

- Denote by \( \Sigma^* \) the set of finite words (strings) over \( \Sigma \), by \( \varepsilon \in \Sigma^* \) the empty word.
- Define \( \overline{\delta} : Q \times \Sigma^* \rightarrow Q \) by
  \[
  \overline{\delta}(q, \varepsilon) = q \quad \text{and} \quad \overline{\delta}(q, wa) = \delta(\overline{\delta}(q, w), a), \quad \text{for all } w \in \Sigma^*, a \in \Sigma.
  \]

- **Language accepted by** \( M \):
  \[ L(M) = \{ w \in \Sigma^* \mid \overline{\delta}(q_0, w) = p, \text{ for some } p \in F \} \]

- **Example:** \( L(M_0) \) is the set of all strings over \( \Sigma = \{ a, b \} \) with an even number of \( b \)'s.

- Gene regulatory networks can be modeled as networks of finite automata.

### Turing machine

Depending on the symbol scanned and the state of the control, in each step the machine

- changes state,
- prints a symbol on the cell scanned, replacing what is written there,
- moves the head left or right one cell.
Formal definition

- \( M = (Q, \Sigma, \Gamma, \delta, q_0, #, F) \)
- \( Q \) is the finite set of states.
- \( \Gamma \) is the finite alphabet of allowable tape symbols.
- \( # \in \Gamma \) is the blank.
- \( \Sigma \subseteq \Gamma \setminus \{#\} \) is the set of input symbols.
- \( \delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\} \) is the next move function (possibly undefined for some arguments)
- \( q_0 \in Q \) is the start state.
- \( F \subseteq Q \) is the set of final (accepting) states.

Recognizing languages

- Instantaneous description: \( \alpha l q \alpha r \), where
  - \( q \) is the current state,
  - \( \alpha_l \alpha_r \in \Gamma^* \) is the string on the tape up to the rightmost nonblank symbol,
  - the head is scanning the leftmost symbol of \( \alpha_r \).
- Move: \( \alpha_l q \alpha_r \vdash \alpha'_l q' \alpha'_r \), by one step of the machine.
- Language accepted by \( M \)

\[ L(M) = \{ w \in \Sigma^* \mid q_0 w \vdash^* \alpha_i q \alpha_r, \text{ for some } q \in F \text{ and } \alpha_i, \alpha_r \in \Gamma^* \} \]

- \( M \) may not halt, if \( w \) is not accepted.

Example

- Turing machine

\[ M = (\{ q_0, \ldots, q_4 \}, \{0,1\}, \{0,1,X,Y,#\}, \delta, q_0, #, \{q_4\}) \]

accepting the language \( L = \{0^n1^n \mid n \geq 1 \} \)

<table>
<thead>
<tr>
<th>( \delta )</th>
<th>0</th>
<th>1</th>
<th>X</th>
<th>Y</th>
<th>#</th>
</tr>
</thead>
<tbody>
<tr>
<td>( q_0 )</td>
<td>( q_1, X, R )</td>
<td>( X )</td>
<td>( Y )</td>
<td>( q_3, Y, R )</td>
<td>( q_2, Y, L )</td>
</tr>
<tr>
<td>( q_1 )</td>
<td>( q_2, 0, R )</td>
<td>( q_2, 0, L )</td>
<td>( q_0, X, R )</td>
<td>( q_2, Y, L )</td>
<td>( q_4, #, R )</td>
</tr>
<tr>
<td>( q_2 )</td>
<td>( q_3 )</td>
<td>( q_2, 0, L )</td>
<td>( q_0, X, R )</td>
<td>( q_2, Y, L )</td>
<td>( q_4, #, R )</td>
</tr>
<tr>
<td>( q_3 )</td>
<td>( q_4 )</td>
<td>( q_3 )</td>
<td>( q_4 )</td>
<td>( q_4 )</td>
<td>( q_4 )</td>
</tr>
</tbody>
</table>

- Example computation

\[
\begin{align*}
q_0011 & \vdash Xq_1011 & \vdash X0q_111 & \vdash Xq_20Y1 & \vdash \\
q_20Y1 & \vdash Xq_30Y1 & \vdash XXq_11 & \vdash XXq_11 & \vdash \\
XXq_2YY & \vdash Xq_2XXY & \vdash XXq_0YY & \vdash XXq_3Y & \vdash \\
XXYYq_4 & \vdash XXYYq_4 & \vdash \\
\end{align*}
\]