Behavior of Automata

Overview

- Automata definition
- Transition graphs
- Language of an automaton
- Equivalence of automata

References

- Robin Milner, “Communication and Concurrency”
- Robin Milner, “Communicating and Mobil Systems”
Automata

Starting point: The components of a system are interacting automata.

An automaton is a quintuple $A = (\Sigma, Q, q_0, \sigma, F)$ with:

- a set $\Sigma$ of actions (sometimes called an alphabet),
- a set $Q = \{q_0, q_1, \ldots\}$ of states,
- a subset $F$ of $Q$ called the accepting states,
- a subset $\sigma$ of $Q \times \Sigma \times Q$ called the transitions,
- a designated start state $q_0$.

A transition $(q, a, q') \in \sigma$ is usually written $q \xrightarrow{a} q'$.

The automaton $A$ is said to be finite if $Q$ is finite.
Transition Graph

An automaton is usually represented by a transition graph, whose nodes are states and whose arcs are transitions.

Example: $A_0$ is a finite automaton over the alphabet $\Sigma = \{a, b, c\}$:

$Q_{A_0} = \{q_0, q_1, q_2, q_3\}$
Language of an Automaton

A string over an alphabet $\Sigma$ is a finite sequence of symbols from that alphabet, usually written to one another and not separated by any characters. If $w$ is a string over $\Sigma$, the length of $w$, written $|w|$, is the number of symbols that it contains. The string of length zero is called the empty string and is denoted by $\varepsilon$.

Let $A$ be an automaton over $\Sigma$, and $s = a_1 \ldots a_n$ a string over $\Sigma$. Then $A$ is said to accept $s$ if there is a path in $A$, from $q_0$ to an accepting state, whose arcs are labeled successively $a_1, \ldots, a_n$. The language of $A$, denoted by $L(A)$, is the set of strings accepted by $A$.

The language $L(A)$ of any finite-state automaton $A$ is regular.
Regular Sets

Operations for building sets of strings:
- **Union**
  \[ S_1 + S_1 = \{ s \mid s \in S_1 \lor s \in S \} \]
- **Concatenation**
  \[ S_1 \cdot S_2 = \{ s_1s_2 \mid s_1 \in S_1, s_2 \in S_2 \} \]
- **Iteration**
  \[ S^* = \{ \varepsilon \} + S + S \cdot S + S \cdot S \cdot S + \ldots \]
  \[ = S^0 + S^1 + S^2 + S^3 + \ldots \]

A set of strings over \( \Sigma \) is said to be regular if it can be built from the empty set \( \emptyset \) and the singleton set \( \{ a \} \) (for each \( a \in \Sigma \)), using just the operations of union, concatenation, and iteration.
Arden’s Rule

The following equation hold:

\[(S_1 \cdot S_2) \cdot S_3 = S_1 \cdot (S_2 \cdot S_3)\]
\[(S_1 + S_2) \cdot T = S_1 \cdot T + S_2 \cdot T\]
\[T \cdot (S_1 + S_2) = T \cdot S_1 + T \cdot S_1\]
\[S \cdot \varepsilon = S\]
\[S \cdot \emptyset = \emptyset\]
\[S \cdot (T \cdot S)^* = (S \cdot T)^* \cdot S\]

Note, \(\emptyset\) means “no path”, whereas \(\varepsilon\) means “empty path”.

**Arden’s Rule:** For any sets of strings \(S\) and \(T\), the equation \(X = S \cdot X + T\) has \(X = S^* \cdot T\) as a solution. Moreover, this solution is unique if \(\varepsilon \not\in S\).
Behavior of Automata

An automaton is deterministic if for each pair \((q, a) \in Q \times \Sigma\) there is exactly one transition \(q \xrightarrow{a} q'\).

Deterministic automata:

Non-deterministic automata:
A tea/coffee vending machine is implemented as a black box with a three-symbol alphabet \( \{25\text{ct}, \text{tea}, \text{coffee}\} \).
Internal Transition Diagrams

Deterministic system S1: Non-deterministic system S2:

Are both systems equivalent?
S1 = S2?

S1:
q0 = 25ct · q1 + ε
q1 = tea · q0 + 25ct · q2
q2 = coffee · q0
q1 = tea · q0 + 25ct · coffee · q0
q0 = 25ct · (tea · q0 + 25ct · coffee · q0) + ε
q0 = 25ct · (tea + 25ct · coffee) · q0 + ε
q0 = (25ct · (tea + 25ct · coffee)) *

S2:
q0 = 25ct · q1 + 25ct · q2 + ε
q1 = tea · q0
q2 = 25ct · q3
q3 = coffee · q0
q2 = 25ct · coffee · q0
q0 = 25ct · tea · q0 + 25ct · coffee · q0 + ε
q0 = 25ct · (tea · q0 + 25ct · coffee · q0) + ε
q0 = 25ct · (tea + 25ct · coffee) · q0 + ε
q0 = (25ct · (tea + 25ct · coffee)) *

The systems S1 and S2 are language-equivalent, but the observable behavior is not the same.
Language Equivalence

- Language equivalence is blind for non-determinism. In fact, each non-deterministic automaton can be transformed into an equivalent deterministic one.

- Language equivalence is blind for deadlocks.

- Language equivalence requires accepting states.
Automata - Summary

Language-equivalence is not suitable for all purposes. If we are interested in interactive behavior, then a non-deterministic automaton cannot correctly be equated behaviorally with a deterministic one.

A different theory is required!