Formal Methods in software development

a.a.2017/2018

Prof.Anna Labella

Operational semantics

- Meaning of program phrases defined in terms of the steps of computation they can take during program execution

The metaphore of abstract machine

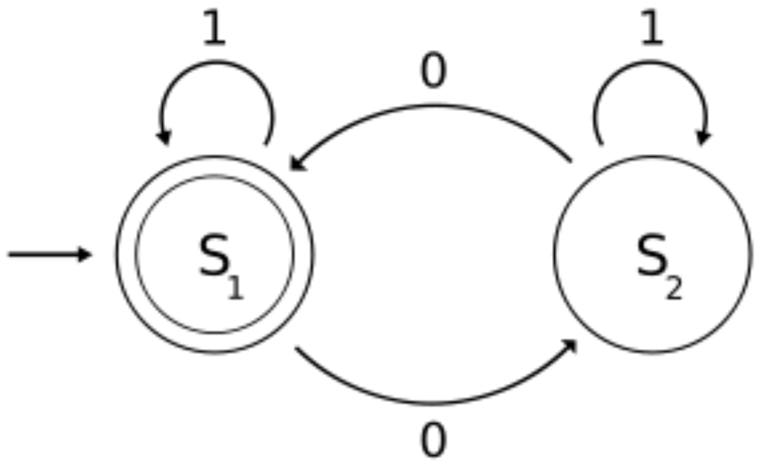
- An automaton is given by a (finite) set of states S; an initial state s_0 and (one or more) terminal one
- Transitions labeled via a finite alphabet Σ
- A transition law δ: $S \times \Sigma \rightarrow S$ (maybe a relation)

Arbib, Michael A. (1969). Theories of Abstract Automata- Prentice Hall

The metaphore of abstract machine

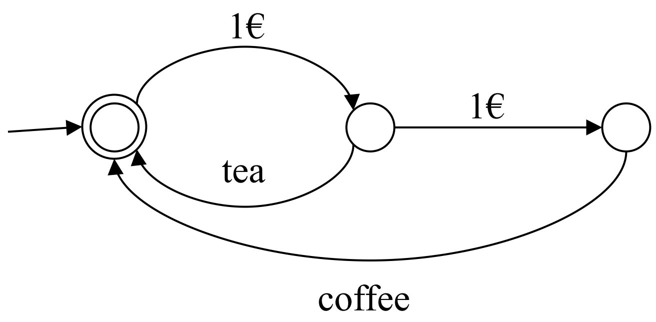
- Automata are characterized through the “accepted” language
- i.e. through the possible sequences of elements from the alphabet going from the initial state to the terminal state

Finite State Automata

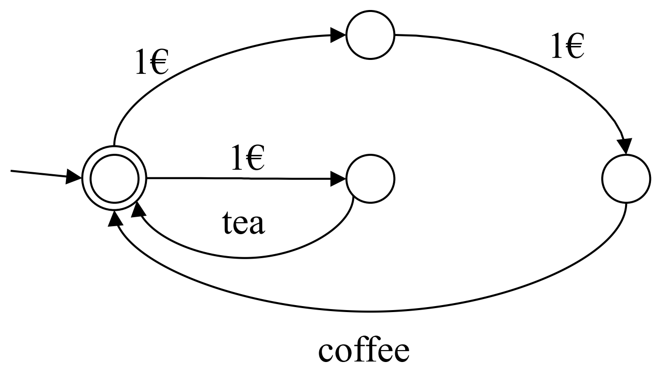


Finite State Automata

- Coffee machine A_1:



- Coffee machine A_2:



- Are the two machines ”the same”?

The algebra of languages

$$<P(\Sigma^*), \subseteq, \cup, \cap, \emptyset, \Sigma^*, \cdot, \{\varepsilon\}>$$

Σ^* is the free monoid generated by the alphabet

$P(\Sigma^*)$ is a boolean algebra, but also a domain

It is a monoid w.r.t. concatenation $\cdot$ (Frobenius product)

The free monoid generated by *A*

$$
\begin{tikzcd}
A \arrow[rr, hook, "i"] \arrow[dr, "\forall f"'] & & L(A) \arrow[dl, "!f"] \\
& M &
\end{tikzcd}
$$

Exercises

Let us take $A = \{1\}$, what is A^* ?

Exercises

Let us take $A = \{1\}$, what is A^* ?

Which kind of coding we get?

Boolean algebra

$$<P(\Sigma^*), \subseteq, \cup, \cap, \emptyset, \Sigma^*>$$

Frobenius product

$$\forall L, M \in \mathcal{P}(\Sigma^*) \quad L \cdot M \overset{def}{=} \{ww' \mid w \in L, w' \in M\}$$

$$\emptyset \cdot L = \emptyset \wedge L \cdot \emptyset = \emptyset$$

$$\{\varepsilon\} \cdot L = L \cdot \{\varepsilon\} = L$$

$$(L \cdot M) \cdot N = L \cdot (M \cdot N)$$

$$L \subseteq L' \quad M \subseteq M' \Rightarrow L \cdot M \subseteq L' \cdot M'$$

Frobenius product distributes over sum (union)

The algebra of languages

We are able to define iteration on $P(\Sigma^*)$

- $L^0 = \{\varepsilon\}$
- $L^{n+1} = L \cdot L^n$
- $L^* = \bigcup_{n \in \mathbb{N}} L^n$

Difficult to axiomatize

Regular expressions

Formulas build from +, $\cdot$ and *, starting from 0, 1 and elements of Σ

Example : $a^* + a \cdot b^*$

Right linear grammars

- Let us have also non terminal (not belonging to Σ) symbols

- A right linear grammar is a set of rewriting rules, where single non terminal elements are rewritten into linear polinomial expressions where elements of the alphabet are left coefficients

Example:

$S \rightarrow aS$ also written $S \rightarrow aS \mid \varepsilon$ or $S = aS + 1$

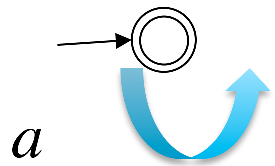
$S \rightarrow \varepsilon$

Theorems

- Languages accepted by finite states automata correspond to regular expressions
- Regular expressions are generated by right linear grammars, which are continuous operators on $P(\Sigma^*)$
- The language associated with a right linear grammar is the minimal solution of a recursive equations system
- We can extend the result to context free grammars (push down automata)

Examples

Let us take the automaton



It accepts the language a^*

defined by the grammar $X = a\ X \mid \varepsilon$

or via the recursive equation $x = ax+1$

as its minimal solution (Tarski's theorem)

More generally

The recursive equation $x = ax+b$

defines a continuous function $P(\Sigma^*) \to P(\Sigma^*)$

and has a^*b as its minimal solution (Tarski's theorem)

In general, every right linear grammar on n non terminal symbols defines a continuous function $P(\Sigma^*)^n \to P(\Sigma^*)^n$

Exercises

Construct regular expressions representing languages, over the alphabet $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$, in which for every string w it holds that:

(i) The number of a's in w is even.

(ii) There are $4i + 1$ b's in w. $(i \geq 0)$

(iii) $|w| = 3i$. $(i \geq 0)$

(i) $(\mathtt{b} \cup \mathtt{c})^* \left(\left(\mathtt{a}(\mathtt{b} \cup \mathtt{c})^* \right)^2 \right)^*$

(ii) $(\mathtt{a} \cup \mathtt{c})^* \mathtt{b} (\mathtt{a} \cup \mathtt{c})^* \left(\left(\mathtt{b}(\mathtt{a} \cup \mathtt{c})^* \right)^4 \right)^*$

(iii) $(\Sigma^3)^*$

We can take more general grammars

grammar $\begin{array}{l} X \to aXa \mid b \\ Y \to Ya^2 \mid bXY \end{array}$ $\Rightarrow$ $\begin{array}{l} X = aXa + b \\ Y = Ya^2 + bXY \end{array}$ equations system

1. $X_0 = 0;$
   $Y_0 = 0;$

2. $X_1 = b;$
   $Y_1 = 0;$

3. $X_2 = aba + b;$
   $Y_2 = 0;$

4. $X_3 = aabaa + aba + b;$
   $Y_3 = 0;$

5. ...

Minimal solution

$X_\infty = a^*ba^*$

$Y_\infty = 0$

Exercises

Show that the following subsets are chain-closed:

(a) $\{\langle x, y\rangle \in D \times D | x \sqsubseteq y\}$, for every cpo D.

(b) $\downarrow(d) := \{x \in D | x \sqsubseteq d\}$ for every d in any cpo D.

(c) $f^{-1}[S] := \{x \in D | f(x) \in S\}$, for every continuous function $f : D \to E$ and chain-closed subset S of E.

(d) $S \cup T$ for every chain-closed subsets S, T of any cpo D.

(e) $\bigcap_{i \in I} S_i$ for every I-indexed family of chain-closed subsets S_i of any cpo D.

(f) $\{\langle x, y\rangle \in D \times D | x = y\}$, for every cpo D.