Team automata : a formal approach to the modeling of collaboration between system components
Beek, M.H. ter

Author: Beek, Maurice H. ter
Title: Team automata : a formal approach to the modeling of collaboration between system components
Issue Date: 2003-12-10

2. Preliminaries

In this chapter we fix most basic notation and terminology used throughout this thesis.

Sets

Set inclusion is denoted by $\subseteq$, whereas proper inclusion is denoted by $\subset$. The set difference of sets V and W is denoted by $V \setminus W$. For a finite set V, its cardinality is denoted by $\#V$. The empty set is denoted by $\varnothing$. For convenience, we sometimes denote the set $\{1, 2, \ldots, n\}$ by $[n]$. Then $[0] = \varnothing$. We sometimes identify a singleton set $\{j\}$ with its only element j.

Let $\mathbb{N}$ denote the set of positive integers. Let $\mathcal{I} \subseteq \mathbb{N}$ be a set of indices given by $\mathcal{I} = \{i_1, i_2, \ldots\}$ with $i_j < i_\ell$ if $1 \leq j < \ell$ and let V_i be a set, for each $i \in \mathcal{I}$. Then $\prod_{i\in\mathcal{I}} V_i$ denotes the cartesian product $\{(v_{i_1}, v_{i_2}, \ldots) \mid v_{i_j} \in V_{i_j}, \text{ for all } j \geq 1\}$. The elements of $\prod_{i\in\mathcal{I}} V_i$ are called vectors. If $\mathcal{I}$ is finite and $\#\mathcal{I} = n$, then the vectors in $\prod_{i\in\mathcal{I}} V_i$ are said to be n-dimensional. Throughout this thesis vectors may be written vertically as well as horizontally. If $v_i \in V_i$, for all $i \in \mathcal{I}$, then $\prod_{i\in\mathcal{I}} v_i$ denotes the element $(v_{i_1}, v_{i_2}, \ldots)$ of $\prod_{i\in\mathcal{I}} V_i$. If $\mathcal{I} = \varnothing$, then $\prod_{i\in\mathcal{I}} V_i = \varnothing$. In addition to the prefix notation $\prod_{i\in\mathcal{I}} V_i$ for a cartesian product, we sometimes also use the infix notation $V_{i_1} \times V_{i_2} \times \cdots$.

Let $j \in \mathcal{I}$. Then $\mathrm{proj}_{\mathcal{I},j} : \prod_{i\in\mathcal{I}} V_i \to V_j$ is the projection function defined by $\mathrm{proj}_{\mathcal{I},j}((a_{i_1}, a_{i_2}, \ldots)) = a_j$. We thus observe that if $\mathcal{I} = \{2, 3\}$, then $\mathrm{proj}_{\mathcal{I},2}((a, b)) = a$. Note moreover that whenever $\mathcal{I} = \mathbb{N}$, then $\mathrm{proj}_{\mathcal{I},j}$ is the standard projection. Similarly, for $J \subseteq \mathcal{I}$, $\mathrm{proj}_{\mathcal{I},J} : \prod_{i\in\mathcal{I}} V_i \to \prod_{i\in J} V_i$ is the projection function defined by $\mathrm{proj}_{\mathcal{I},J}(a) = \prod_{j\in J} \mathrm{proj}_{\mathcal{I},j}(a)$. Whenever $\mathcal{I}$ is clear from the context we write proj_j and proj_J rather than $\mathrm{proj}_{\mathcal{I},j}$ and $\mathrm{proj}_{\mathcal{I},J}$. Note that for each $j \in \mathcal{I}$ and $a \in \prod_{i\in\mathcal{I}} V_i$ we have $\mathrm{proj}_{\{j\}}(a) = \prod_{j\in\{j\}} \mathrm{proj}_j(a)$, which we do not identify with $\mathrm{proj}_j(a)$. Formally, we have $\mathrm{proj}_j(\mathrm{proj}_{\{j\}}(a)) = \mathrm{proj}_j(a)$.

The set $\{V_i \mid i \in \mathcal{I}\}$ is said to form a partition (of $\bigcup_{i\in\mathcal{I}} V_i$) if the V_i are pairwise disjoint, nonempty sets.

Functions

All functions considered are total, unless explicitly stated otherwise.

Let $f : A \to A'$ and let $g : B \to B'$ be functions. Then $f \times g : A \times B \to A' \times B'$ is defined as $(f \times g)(a, b) = (f(a), g(b))$. We will use $f^{[2]}$ as shorthand notation for $f \times f$. Thus $f^{[2]}(a, b) = (f(a), f(b))$. This notation should not be confused with iterated function application. In particular, we will use ${\mathrm{proj}_{\mathcal{I},j}}^{[2]}$ as shorthand notation for $\mathrm{proj}_{\mathcal{I},j} \times \mathrm{proj}_{\mathcal{I},j}$ and likewise ${\mathrm{proj}_{\mathcal{I},J}}^{[2]}$ for $\mathrm{proj}_{\mathcal{I},J} \times \mathrm{proj}_{\mathcal{I},J}$. We write ${\mathrm{proj}_j}^{[2]}$ and ${\mathrm{proj}_J}^{[2]}$ rather than ${\mathrm{proj}_{\mathcal{I},j}}^{[2]}$ and ${\mathrm{proj}_{\mathcal{I},J}}^{[2]}$ whenever $\mathcal{I}$ is clear from the context. If $C \subseteq A$, then $f(C) = \{f(a) \mid a \in C\}$. Thus if $D \subseteq A \times A$, then $f^{[2]}(D) = \{(f(d_1), f(d_2)) \mid (d_1, d_2) \in D\}$.

The function f is injective if $f(a_1) \neq f(a_2)$ whenever $a_1 \neq a_2$, f is surjective if for every $a' \in A'$ there exists an $a \in A$ such that $f(a) = a'$, and f is a bijection if f is injective and surjective. The restriction of the function f to a subset C of its domain A is denoted by $f \restriction C$ and is defined as the function $C \to A'$ defined by $(f \restriction C)(c) = f(c)$, for all $c \in C$.

Alphabets, Words, Languages

An alphabet is a set of letters — symbols — which may be used, e.g., to represent actions of systems. We do not impose any a priori constraints on the size of an alphabet. Alphabets may thus be empty and they may be infinite. For the remainder of this chapter we let Σ be an arbitrary but fixed alphabet.

A word (over Σ) is a sequence of symbols (from Σ). A word may be a finite or infinite sequence of symbols, resulting in finite and infinite words, respectively. An infinite word is also referred to as an ω-word. The empty sequence is called the empty word and denoted by λ. As usual we represent nonempty words $a_1, a_2, \ldots$ over Σ as strings $a_1 a_2 \cdots$. For a finite word w, we use the notation $|w|$ to denote its length. Thus $|\lambda| = 0$ and if $w = a_1 a_2 \cdots a_n$, with $n \geq 1$ and $a_i \in \Sigma$, for all $1 \leq i \leq n$, then $|w| = n$.

Words may also be considered as functions which assign symbols to positions. Thus a finite word $w = a_1 a_2 \cdots a_n$, with $n \geq 1$ and $a_i \in \Sigma$ for all $1 \leq i \leq n$, is identified with the function $w : [n] \to \Sigma$ defined by $w(i) = a_i$, for all $1 \leq i \leq n$. Similarly, an infinite word $w = a_1 a_2 \cdots$, with $a_i \in \Sigma$ for all $i \geq 1$, defines the function $w : \mathbb{N} \to \Sigma$ by $w(i) = a_i$, for all $i \geq 1$. To the empty word λ we associate the function $\lambda : [0] \to \Sigma$, which has an empty domain.

For a finite word w over Σ and a symbol $a \in \Sigma$, we use $\#_a(w)$ to denote the number of occurrences of a in w. Thus $\#_a(w) = \#\{i \in [|w|] \mid w(i) = a\}$. Note that $\#_a(\lambda) = 0$, for all a. For a (finite or infinite) word w, its alphabet, denoted by alph(w), consists of all symbols that actually occur in w. Thus alph(w) $= \{a \in \Sigma \mid \exists i \in \mathbb{N} : \; w(i) = a\}$. Note that alph($\lambda$) $= \varnothing$ and that alph(w) may be an infinite set if Σ is infinite and w is an infinite word.

The set of all finite words over Σ (including λ) is denoted by Σ^*. The set $\Sigma^+ = \Sigma^* \setminus \{\lambda\}$ consists of all nonempty finite words. By convention $\Sigma \subseteq \Sigma^+$. The set of all infinite words over Σ is denoted by Σ^ω. By Σ^∞ we denote the set of all words over Σ. Thus $\Sigma^\infty = \Sigma^* \cup \Sigma^\omega$. A language (over Σ) is a set of words (over Σ). A language consisting solely of finite words is called finitary. If $L \subseteq \Sigma^\omega$, i.e. all words of L are infinite, then L is called an infinitary language or ω-language. As usual we refer to a collection (set) of languages as a family of languages.

Concatenation

Using the operation of concatenation, two words (over Σ) are combined into one word (over Σ) by gluing them together.

Formally, given $u, v \in \Sigma^\infty$, their concatenation $u \cdot v$ is defined as follows. If $u, v \in \Sigma^*$, then $u \cdot v(i) = u(i)$ for $i \in [|u|]$ and $u \cdot v(|u| + i) = v(i)$ for $i \in [|v|]$. Note that $|u \cdot v| = |u| + |v|$. If $u \in \Sigma^*$ and $v \in \Sigma^\omega$, then $u \cdot v(i) = u(i)$ for $i \in [|u|]$ and $u \cdot v(|u| + i) = v(i)$ for $i \geq 1$. If $u \in \Sigma^\omega$ and $v \in \Sigma^\infty$, then $u \cdot v(i) = u(i)$ for all $i \geq 1$. In the last two cases $u \cdot v \in \Sigma^\omega$. Note that $u \cdot \lambda = \lambda \cdot u = u$, for all $u \in \Sigma^\infty$. Since concatenation is associative this implies that Σ^∞ with concatenation and unit element λ is a monoid. Moreover, since concatenation of two finite words yields a finite word, also Σ^* with concatenation restricted to Σ^* is a monoid with unit element λ.

The concatenation of two languages K and L (over Σ) is the language $K \cdot L$ (over Σ) defined by $K \cdot L = \{u \cdot v \mid u \in K, \; v \in L\}$. Observe that $K \cdot L$ is finitary if and only if both K and L are finitary. Moreover, $K \cdot L = K$ if $L = \{\lambda\}$ or K is infinitary. In the sequel, we will mostly write uv and KL rather than $u \cdot v$ and $K \cdot L$, respectively.

For $u \in \Sigma^\infty$ we set $u^0 = \lambda$ and $u^{n+1} = u^n \cdot u$, for all $n \geq 0$. Note that if $u \in \Sigma^\omega$, then $u^n = u$, for all $n \geq 1$. Similarly, for a language $K \subseteq \Sigma^\infty$ we have $K^0 = \{\lambda\}$ and $K^{n+1} = K^n \cdot K$, for all $n \geq 0$.

Prefixes

A word $u \in \Sigma^*$ is said to be a (finite) prefix of a word $w \in \Sigma^\infty$ if there exists a $v \in \Sigma^\infty$ such that $w = uv$. In that case we write $u \leq w$. If $u \leq w$ and $u \neq w$, then we may use the notation $u < w$. Moreover, if $|u| = n$, for some $n \geq 0$, then u is said to be the prefix of length n of w, denoted by $w[n]$. Note that $w[0] = \lambda$. The set of all prefixes of a word w is denoted by

$\mathrm{pref}(w)$ and it is defined as $\mathrm{pref}(w) = \{u \in \Sigma^* \mid u \leq w\}$. Note that $\mathrm{pref}(w)$ is finite if and only if $w \in \Sigma^*$. Note also that, for a word $x \in \Sigma^\infty$, whenever $\mathrm{pref}(w) = \mathrm{pref}(x)$, then $w = x$.

For a language K, $\mathrm{pref}(K) = \bigcup\{\mathrm{pref}(w) \mid w \in K\}$. Thus $K \subseteq \mathrm{pref}(K)$ whenever K is a finitary language. A language K is prefix closed if and only if $K \supseteq \mathrm{pref}(K)$. A family of languages L is prefix closed if $\mathrm{pref}(K) \in \mathsf{L}$ for all $K \in \mathsf{L}$.

Limits

Both finite and infinite words can be defined as limits of their prefixes. Let $v_1, v_2, \dots \in \Sigma^*$ be an infinite sequence of words such that $v_i \leq v_{i+1}$, for all $i \geq 1$. Then $\lim_{n\to\infty} v_n$ is the unique word $w \in \Sigma^\infty$ defined by $w(i) = v_j(i)$, for all $i, j \in \mathbb{N}$ such that $i \leq |v_j|$. Thus $v_i \leq w$ for all $i \geq 1$ and $w = v_k$ whenever there exists a $k \geq 1$ such that $v_n = v_{n+1}$ for all $n \geq k$. For a word $u \in \Sigma^\infty$ we define $u^\omega = \lim_{n\to\infty} u^n$ if $u \in \Sigma^*$ and $u^\omega = u$ if $u \in \Sigma^\omega$. Note that $\lambda^\omega = \lambda$. For an infinite sequence $u_1, u_2, \ldots \in \Sigma^\infty$ we define the word $u_1 \cdot u_2 \cdot \cdots \in \Sigma^\infty$ by $u_1 \cdot u_2 \cdot \cdots = \lim_{n\to\infty} u_1 \cdot u_2 \cdot \cdots \cdot u_n$ if $u_i \in \Sigma^*$, for all $i \geq 1$, and $u_1 \cdot u_2 \cdot \cdots = u_1 \cdot u_2 \cdot \cdots \cdot u_{n-1} \cdot u_n$ if $u_n \in \Sigma^\omega$, for some $n \geq 1$.

These notations are carried over to languages in the natural way: for $K, K_1, K_2, \ldots \subseteq \Sigma^\infty$, we set $K^\omega = \{u_1 u_2 \cdots \mid u_i \in K, \text{ for all } i \geq 1\}$ and $K_1 \cdot K_2 \cdot \cdots = \{u_1 u_2 \cdots \mid u_i \in K_i, \text{ for all } i \geq 1\}$. Observe that $\Sigma^\omega = \{a_1 a_2 \cdots \mid a_i \in \Sigma, \text{ for all } i \geq 1\}$ is indeed the set consisting of all infinite words over Σ.

Homomorphisms

Let $h : \Sigma \to \Gamma^*$ be a function assigning to each letter of Σ a finite word over the alphabet Γ. The homomorphic extension of h to Σ^*, also denoted by h, is defined in the usual way by $h(\lambda) = \lambda$ and $h(xy) = h(x)h(y)$ for all $x, y \in \Sigma^*$. This homomorphism is further extended to Σ^∞ by setting $h(\lim_{n\to\infty} v_n) = \lim_{n\to\infty} h(v_n)$, for all $v_1, v_2, \ldots \in \Sigma^*$ such that for all $i \geq 1$, $v_i \leq v_{i+1}$. Note that this is well defined, since $v_i \leq v_{i+1}$ implies $h(v_i) \leq h(v_{i+1})$. Note however that if h is erasing, i.e. $h(a) = \lambda$ for some $a \in \Sigma$, then there exists a word $x \in \Sigma^\omega$ such that $h(x) \in \Sigma^*$. For such x we have $h(xy) = h(x)$, for all $y \in \Sigma^\infty$, and consequently $h(xy) = h(x)h(y)$ is no longer guaranteed. In fact, $h(xy) = h(x)h(y)$, for all $x, y \in \Sigma^\infty$, if and only if either h is not erasing or $h(a) = \lambda$, for all $a \in \Sigma$. Thus $h : \Sigma \to \Gamma^*$ cannot always be lifted to a homomorphism on Σ^∞. Still we sometimes abuse terminology and refer to the extension $h : \Sigma^\infty \to \Gamma^\infty$ of h as a homomorphism. If $h(\Sigma) \subseteq \Gamma$, then we refer to h as a coding, and if $h(\Sigma) \subseteq \Gamma \cup \{\lambda\}$, then h is called a weak coding.

The function $\text{pres}_{\Sigma,\Gamma} : \Sigma \to \Gamma^*$, defined by $\text{pres}_{\Sigma,\Gamma}(a) = a$ if $a \in \Gamma$ and $\text{pres}_{\Sigma,\Gamma}(a) = \lambda$ otherwise, preserves the symbols from Γ and erases all other symbols. Whenever Σ is clear from the context, we simply write pres_{Γ} rather than $\text{pres}_{\Sigma,\Gamma}$. Note that $\text{pres}_{\Sigma,\Gamma}$ is a weak coding.