Contents

Chapter 7

Infinite Traces

Paul Gastin
Université Paris 6
LITP, Institut Blaise Pascal
4, place Jussieu
F-75252 Paris Cédex 05

Antoine Petit
Université Paris Sud
LRI, URA CNRS 410
Bât. 490
F-91405 Orsay Cédex

Supported by the ESPRIT Basic Research Actions No. 6067: Algebraic and Syntactic Methods in Computer Science (ASMICS 2) and No. 6317: Causal Calculi Based on Nets (CALIBAN).

7.1 Introduction

Finite traces were introduced by A. Mazurkiewicz as a semantic model for non-sequential processes. The motivation for an infinitary extension of traces is twofold.

First, some non-sequential systems are not designed to terminate, e.g. distributed operating systems. In order to abstract their behaviors, the notion of infinite traces is clearly needed. Since an infinite behavior may be seen as the *limit* of a sequence of finite behaviors, the set of infinite traces has to be the *completion* of the set of finite traces. In other words, each infinite trace must be the limit of a sequence of finite traces and, conversely, each "good" sequence of finite traces must converge to an infinite trace.

Second, some programming structures do not admit convenient semantics within the monoid of finite traces. For instance, behaviors of recursive programs are frequently described by solutions of fix-point equations of the form $x = f(x)$. Depending on the mapping f, different techniques may be used to obtain these fix-points.

- Assume first that the mapping f is non decreasing with respect to some partial order on the semantic domain. Iterating the mapping f starting from the least element $\perp$ of the semantic domain, we obtain a non decreasing sequence $(f^n(\perp))_{n\geq 0}$. If the mapping f is continuous and if the semantic domain is complete with respect to this partial order, this non decreasing sequence admits a least upper bound which is the least fix-point of f.

- Assume now that the mapping f is contracting with respect to some metric on the semantic domain. The sequence $(f^n(x))_{n\geq 0}$ is Cauchy for any starting point

x. If the semantic domain is complete as a metric space, this sequence converges to the unique fix-point of f.

For both techniques, complete semantic domains are needed. Unfortunately, the set of finite traces is not complete, neither with respect to the prefix order nor with respect to any semantically satisfactory metric (see Section 7.5.3). Therefore, both an order and a metric completion of the set of finite traces must be introduced. This gives rise to different notions of infinite traces.

The order completion of the set of finite traces defines the set of infinite *real* traces where a real trace may be identified with an ideal (a prefix-closed and directed set) of finite traces (see Sections 7.2 and 7.3). Note that, when a semantic domain is not complete, fix-point equations are frequently solved in the powerset of the domain. Solutions are then directed sets which may be identified with elements of the order completion of the domain. Hence, the two approaches (solving fix-point equations in the powerset of a semantic domain on one hand, and in the order completion of the domain on the other hand) are closely related. From this point of view, infinite real traces may simply be seen as convenient abstractions of directed sets of finite traces.

Unfortunately, no metric on the set of real traces is fully satisfactory. In order to obtain a *good* metric completion of the set of finite traces, a slight generalization of real traces, namely complex traces, was introduced by V. Diekert (see Sections 7.4 and 7.5).

In this chapter, we also introduce transfinite traces as transfinite generalizations of dependence graphs (see Chapter 1). The semantic motivation for transfinite traces may seem more difficult. This is due to the implicit assumption that there is a positive lower bound to the execution time of all actions. If this restriction is removed, an infinite process may be executed within a finite amount of time. For instance, being inspired by oscillating mechanisms, one may consider an infinite process where each step takes half the time taken by the previous step. Such a process terminates within a finite amount of time and may be followed by another activity. The set of transfinite traces is the natural semantic domain for such situations. Apart from this semantic motivation, transfinite traces are also useful because they allow a uniform framework for the definition of both real and complex traces. Indeed, the set of real traces is embedded in the set of transfinite traces while the set of complex traces is a quotient of the set of transfinite traces.

Real and transfinite traces are defined in Section 7.2. Their basic properties are also investigated in this section. The set of transfinite traces equipped with a natural concatenation forms a left-cancellative monoid. On the other hand, the set of real traces is only a partial monoid since the concatenation of two real traces may give a transfinite (non-real) trace. As a basic tool, products of infinite (and even transfinite) sequences of traces are defined. In particular, these products allow us to relate infinite words and infinite traces and to state a Foata normal form theorem for infinite traces.

In Section 7.3, we study the properties of the sets of real traces and of transfinite traces with respect to the prefix order. We are mainly interested in the *completeness* and the *algebraicity* of these posets (partially ordered sets). More precisely, we prove that the set of real traces is coherently complete (every coherent set of real traces admits a least upper bound) whereas the set of transfinite traces is only bounded complete (every bounded set of transfinite traces admits a least upper bound). Moreover, we show that the set of real traces and the set of transfinite traces are both algebraic:

every real trace (transfinite trace resp.) is the least upper bound of a set of compact real traces (compact transfinite traces resp.). Finally, we prove that a real trace is compact if and only if it is a finite trace. Altogether, this shows that the set of real traces is a Scott domain which is precisely the prefix order completion of the set of finite traces.

As it turns out, however, the set of real traces is not fully satisfactory with respect to the concatenation which is only partially defined. Moreover, although the set of real traces may be provided with metrics which make it a complete metric space, the concatenation (even restricted to finite traces) is not continuous for these metrics. Complex traces are introduced in order to obtain a metric completion of the set of finite traces equipped with a uniformly continuous concatenation.

In Section 7.4, the monoid of complex traces is defined as a quotient of the monoid of transfinite traces by a suitable congruence. Basic properties of the concatenation and of infinite products on complex traces are investigated. In fact, complex traces are simply a slight generalization of real traces: the set of real traces is embedded in the set of complex traces and a complex trace may be identified with a real trace together with an alphabetic information.

The topological properties of complex traces are studied in Section 7.5. The set of complex traces equipped with a natural ultrametric is a complete and compact metric space. For the topology induced by this metric, the set of finite traces is an open, discrete and dense subset of the set of complex traces. Moreover, the concatenation on complex traces is uniformly continuous. In other words, the set of complex traces is the metric completion of the set of finite traces and the concatenation on complex traces is the unique continuous extension of the concatenation on finite traces.

Among all the infinite trace languages describing the behaviors of non-sequential systems, rational and recognizable languages play a central role. Indeed, as in any monoid used as a semantic domain, rational languages describe the behaviors of systems which can be built in a modular way from a finite set of very simple systems. On the other hand, recognizable languages represent the behaviors of finite state systems. Recall that in the case of word languages (i.e. in the case of sequential systems), Kleene's and Büchi's theorems state that the two families coincide: a word language is rational if and only if it is recognizable. Concerning finite traces (i.e. in the case of finite non-sequential systems), the family of recognizable languages is included in the family of rational languages. Moreover, the inclusion is strict as soon as the independence relation is not empty. Nevertheless, Ochmański's theorem (Chapter 4) asserts that a language of finite traces is recognizable if and only if it is concurrent rational. The remainder of this Chapter is devoted to the study of the families of rational and recognizable infinite trace languages and in particular to a generalization of Kleene's, Büchi's and Ochmański's theorems to infinite trace languages.

In Section 7.6, we introduce and study the rational languages of infinite traces. Formally, the family of rational languages is defined as the smallest family which contains the finite sets of finite traces and which is closed under union, concatenation, $*$-iteration and ω-iteration. Depending on the concatenation and the ω-product used, either the family of rational real trace languages or the family of rational complex trace languages is obtained. The main results of this section are normal forms for rational sets of real traces and of complex traces. We also give the links between rational real trace languages and rational word languages on one hand and between rational complex trace languages and rational real trace languages on the other hand. Apart from these characterizations, we prove that the intersection, the left quotient and the right quotient of a rational set by a recognizable set are rational too.

We introduce and study in Section 7.7 the family of recognizable languages of real traces. These languages are defined by extending in a natural way the definition of recognizable word languages using recognizing morphisms. We prove that the family of recognizable real trace languages is closed under union, intersection, complement and (real) concatenation. These results together with the closure by iterations restricted to connected languages yield a normal form theorem for recognizable real trace languages. Next, finite and infinite concurrent iterations on real trace languages are defined. It turns out that the family of recognizable real trace languages is closed under these new operations. Then the main result of this section states that the family of recognizable real trace languages coincides with the family of concurrent rational real trace languages. This last family is defined as the family of rational real trace languages by simply replacing the classical iterations by their concurrent versions. Besides, we prove that the family of recognizable real trace languages is closed under left and right quotients by arbitrary real trace languages.

Finally, we deal with recognizable complex trace languages in Section 7.8. First we give the links between recognizable real trace languages and recognizable complex trace languages. We then prove that the family of recognizable complex trace languages is closed under union, intersection, complement, concatenation, quotients by arbitrary languages and iterations restricted to connected languages. Next, we obtain a normal form theorem for recognizable complex trace languages which generalizes the corresponding result on recognizable real trace languages. Finally, we extend Kleene's and Büchi's theorems on words and Ochmański's theorem on finite traces by proving that the family of recognizable complex trace languages coincides with the family of concurrent rational complex trace languages, i.e. the smallest family of complex trace languages which contains the finite sets of finite traces and which is closed under union, concatenation, finite and infinite concurrent iterations.

7.2 Real and Transfinite Traces

In this section, we introduce *real* and *transfinite* traces which are infinitary generalizations of finite traces. Although it would be possible to introduce directly an equivalence relation on ω-words (transfinite words resp.) in order to define real traces (transfinite traces resp.), we prefer to use the framework of dependence graphs (see Chapter 1). This approach provides a much more natural extension of finite traces and allows to define uniformly real traces, transfinite traces and complex traces (defined in Section 7.4).

The notion of ordinal will be extensively used throughout. We refer to [Ros82] for a complete course on ordinals. In fact, we will use very few results from ordinal theory. Mainly, we will use ordinals to number sequences. In order to fix notations, we recall that the finite ordinals are the natural numbers, the first non-finite ordinal is denoted by ω and the first non-countable ordinal is denoted by $\aleph_1$.

7.2.1 Definition

Let Σ be a finite alphabet and let $D \subseteq \Sigma \times \Sigma$ be a reflexive and symmetric relation over Σ, called the dependence relation. Its complement $I = \Sigma \times \Sigma \setminus D$ is the independence relation. The pair (Σ, D) is called the dependence alphabet. Throughout this chapter, we will depict a dependence alphabet by its graph (without reflexive edges). For instance, the dependence alphabet (Σ, D) defined by $\Sigma = \{a, b, c, d, e\}$ and $D = \{(a,a),(b,b),(c,c),(d,d),(e,e),(a,b),(b,a),(b,c),(c,b),(d,e),(e,d)\}$ will be represented by the following graph:

$$(\Sigma, D) = a \text{ --- } b \text{ --- } c \qquad d \text{ --- } e$$

Throughout this section, all examples will be given with respect to this dependence alphabet.

A dependence graph will be defined as an isomorphism class of labelled directed graphs. Recall that two labelled graphs $[V_1, E_1, \lambda_1]$ and $[V_2, E_2, \lambda_2]$ are said to be isomorphic if there exists a bijection $\xi : V_1 \longrightarrow V_2$ between their sets of vertices which preserves the edges ($(p,q) \in E_1 \iff (\xi(p), \xi(q)) \in E_2$) and the labelling ($\lambda_1(p) = \lambda_2(\xi(p))$). For the sake of simplicity, we will often identify a concrete labelled graph and its isomorphism class.

Definition 7.2.1 *A* dependence graph *over the dependence alphabet* (Σ, D) *is an isomorphism class of labelled directed acyclic graphs* $[V, E, \lambda]$, *where the set* V *of vertices is countable,* $E \subseteq V \times V$ *is the set of edges,* $\lambda : V \to \Sigma$ *is the labelling function and it holds*

- *edges are between dependent vertices: for all* $p, q \in V$,

$$(\lambda(p), \lambda(q)) \in D \iff p = q \text{ or } (p,q) \in E \text{ or } (q,p) \in E$$

- *the induced partial order* $\preceq = E^*$ *(the reflexive and transitive closure of the edge relation* E*) is well-founded: there is no infinite strictly decreasing sequence of vertices* $p_0 \succ p_1 \succ p_2 \succ \cdots$

The set of dependence graphs is denoted by $\mathbb{G}(\Sigma, D)$, *or simply by* $\mathbb{G}$. *A dependence graph is also called a (transfinite) trace. The set of finite dependence graphs (finite traces) is denoted by* $\mathbb{M}(\Sigma, D)$, *or simply by* $\mathbb{M}$.

This is a very natural generalization of the definition of finite dependence graphs presented in Chapter 1. We simply allow the set of vertices to be countable and we require in addition that the reflexive and transitive closure of the edge relation is well-founded. This restriction ensures that every dependence graph admits a finite set of initial actions.

Moreover, for each dependence graph $g = [V, E, \lambda]$ and each letter $a \in \Sigma$, this last restriction implies that the set $V_a = \lambda^{-1}(a) = \{p \in V \mid \lambda(p) = a\}$ of vertices with label a is a *well-ordered* subset of V, i.e., the restriction of the induced partial order $\preceq$ to V_a is a well-founded *total* order. We denote by $|g|_a$ the number of occurrences of a in g, i.e., $|g|_a$ is the countable ordinal associated with the well-ordered set V_a. Hence, every dependence graph g admits a standard representation where its set of vertices is

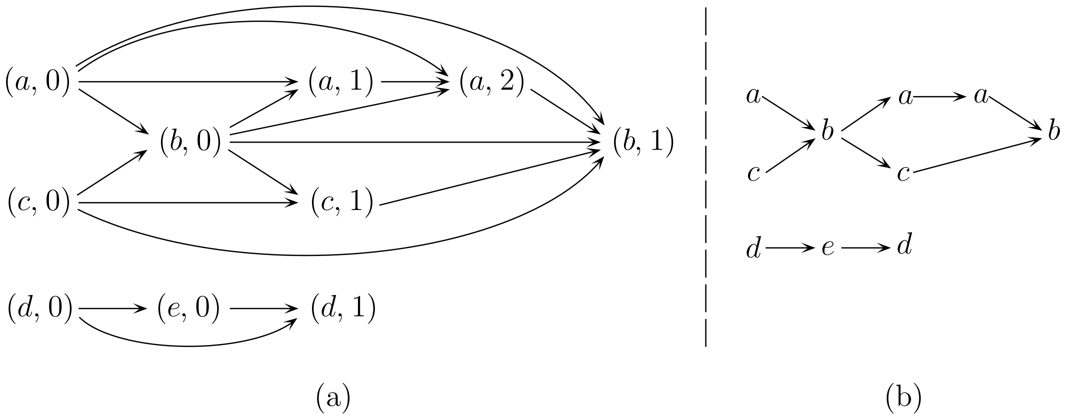

Figure 7.1: A *finite* dependence graph

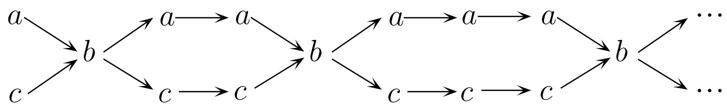

Figure 7.2: A *real* dependence graph

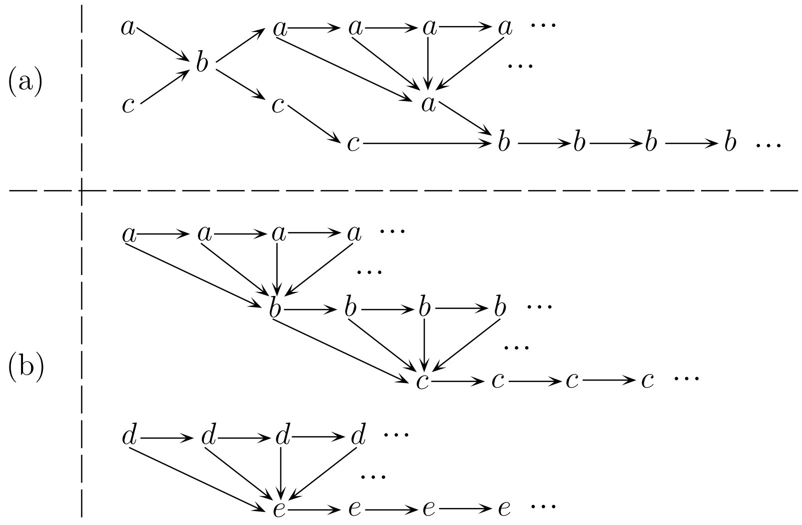

Figure 7.3: Two *transfinite* dependence graphs

$V = \{(a,i) \mid a \in \Sigma \text{ and } i < |g|_a\}$ and the labelling is trivially defined by $\lambda((a,i)) = a$ for all $(a,i) \in V$.

For instance, the standard representation of a finite dependence graph is presented in Figure 7.1 (a). Since a dependence graph is an isomorphism class of graphs, we will only write the labels of the vertices. Still in order to simplify the pictures, we will not necessarily draw the edges which can be obtained by composition of other edges. Using these remarks, the dependence graph of Figure 7.1 (a) can be simply presented by its *Hasse diagram* as in Figure 7.1 (b). Finally, three infinite dependence graphs are presented in Figures 7.2 and 7.3.

Let $g = [V, E, \lambda]$ be a dependence graph and $p \in V$ be a vertex of g. The *past* of p for the induced partial order $\preceq\, = E^*$ is the set $\downarrow_g p = \{q \in V \mid q \preceq p\}$. When no confusion may arise, the past of p will be simply denoted by $\downarrow p$. For instance, in the graph of Figure 7.1 we have $\downarrow(a,1) = \{(a,0),(a,1),(b,0),(c,0)\}$. Note that the infinite graph of Figure 7.2 satisfies a stronger property than merely the well-foundedness of its induced partial order: all its vertices have a finite past. On the other hand, in the graph of Figure 7.3 (b), the vertices labelled by b, c or e have an infinite past. This important remark leads to the following definition.

Definition 7.2.2 *Let $g = [V, E, \lambda] \in \mathbb{G}(\Sigma, D)$ be a dependence graph.*

The real part $\mathrm{Re}(g)$ *and the* transfinite part $\mathrm{Tr}(g)$ *of g are the restrictions of the graph g to the sets of vertices* $\{p \in V \mid \downarrow p \text{ is finite}\}$ *and* $\{p \in V \mid \downarrow p \text{ is infinite}\}$ *respectively.*

A real trace *(or* real *dependence graph) is a dependence graph with an empty transfinite part. The set of real traces will be denoted by $\mathbb{R}(\Sigma, D)$, or simply by $\mathbb{R}$.*

The alphabet *of g,* $\mathrm{alph}(g) = \lambda(V)$*, is the set of letters which occur in g. The* alphabet at infinity *of g is* $\mathrm{alphinf}(g) = \{a \in \Sigma \mid \lambda^{-1}(a) \text{ is infinite}\} \cup \mathrm{alph}(\mathrm{Tr}(g))$.

Note that $\mathrm{Re}(g)$ and $\mathrm{Tr}(g)$ are dependence graphs. Moreover, $\mathrm{Re}(g)$ is clearly a real trace. This is a direct consequence of the following easy fact.

Remark 7.2.3 The restriction of a dependence graph $g = [V, E, \lambda]$ to any subset of V is a dependence graph too.

For instance, the graphs of Figures 7.1 and 7.2 are real whereas the graphs of Figure 7.3 are not real. Their real parts and transfinite parts are presented in Figure 7.4. The alphabets of the graphs of Figures 7.1–7.3 are respectively Σ, $\{a,b,c\}$, $\{a,b,c\}$ and Σ. Their alphabets at infinity are respectively $\emptyset$, $\{a,b,c\}$, $\{a,b\}$ and Σ.

We end this subsection with some semantic considerations concerning real and non-real dependence graphs. A very natural point of view in computer science is to assume that there exist a uniform lower bound and a uniform upper bound on the execution time of all actions. Then real graphs are exactly those graphs in which each action may be executed within a finite amount of time. This is probably the main reason for which real traces may be considered as the most semantically significant infinite traces. Moreover, we will see throughout the chapter that real traces possess very good properties.

Therefore, one may wonder why we do not restrict ourselves to the study of real traces. There are in fact three main reasons for this. First, we will see in the next subsection that only a partial concatenation is defined on real traces. Hence, the algebraic structure of transfinite traces or complex traces (defined in Section 7.4) is