INFORMATIQUE THÉORIQUE ET APPLICATIONS

WALTER VOGLER

A generalization of traces

Informatique théorique et applications, tome 25, n° 2 (1991), p. 147-156.

<http://www.numdam.org/item?id=ITA_1991__25_2_147_0>

© AFCET, 1991, tous droits réservés.

L'accès aux archives de la revue « Informatique théorique et applications » implique l'accord avec les conditions générales d'utilisation (http://www.numdam.org/legal.php). Toute utilisation commerciale ou impression systématique est constitutive d'une infraction pénale. Toute copie ou impression de ce fichier doit contenir la présente mention de copyright.

NUMDAM

Article numérisé dans le cadre du programme
Numérisation de documents anciens mathématiques
http://www.numdam.org/

Informatique théorique et Applications/Theoretical Informatics and Applications
(vol. 25, n° 2, 1991, p. 147 à 156)

A GENERALIZATION OF TRACES (*)

by Walter VOGLER (1)

Communicated by Wilfried BRAUER

Abstract. – *Traces describe the behaviour of distributed systems as labelled partial orders under the assumption that actions accessing the same object cannot occur simultaneously. We drop this assumption and show that for the resulting generalized traces will still have some results that intuitively justify the use of traces, namely: Traces corresponds to equivalence classes of global observations, and they correspond to tuples of local observations.*

Résumé. – *Les traces décrivent le comportement des systèmes distribués comme ordres partiels étiquetés sous l'hypothèse que des actions accédant à la même ressource ne peuvent se produire simultanément. Nous supprimons cette restriction et nous montrons que pour les traces généralisées ainsi définies, on retrouve divers résultats qui justifient intuitivement l'usage des traces. En particulier, les traces correspondent à des classes d'équivalence d'observations globales et elles correspondent à des t-uplets d'observations locales.*

1. INTRODUCTION

On some level of abstraction, executions of sequential systems can be described as sequences of elementary actions from some given set. How executions of concurrent or distributed systems should be described on this level is not that clear: One could still take sequences and say concurrency of two actions is expressed by allowing their occurence in arbitrary order. This interleaving approach identifies concurrency with some form of nondeterminism. This is certainly not always adequate: If two tasks can be executed in any order this does not mean that they can be executed simultaneously. A more appropriate approach that takes at least some concurrency into account is to take step sequences, where a step is a multiset of actions that occur in parallel.

(*) Received November 1986, revised July 1990.
This work was partially supported by the ESPRIT Basic Research Action No. 3148 DEMON (Design Methods Based on Nets).

(1) Institut für Informatik, Technische Universität München, Arcisstr. 21, D-8000 München 2, R.F.A.

Informatique théorique et Applications/Theoretical Informatics and Applications
0988-3754/91/02 147 10/$3.00/© AFCET-Gauthier-Villars

A third possibility is to describe an execution by a partial order of action occurrences, which nicely reflects the distributedness of the system. An example of this approach is the theory of traces in the sense of Mazurkiewicz, see e. g. [1, 7, 8]; trace monoids as a mathematical structure were already studied in [3] and given the name free partially commutative monoids in [6].

In trace theory it is assumed that a static dependence relation on actions is given; a possible interpretation is that dependent actions access the same object, thus it is important in which order such actions occur. A trace is a partial order of action occurences, where occurrences of dependent actions are always ordered while occurrences of independent actions are not directly ordered, *i. e.* cannot be immediate predecessors of each other. Of course, if independent actions, a, b are both dependent to c, a occurs before c, and c occurs before b, then a occurs before b by transitivity.

What makes traces so appealing are the following correspondences, which are shown in [7]. On the one hand, traces correspond to equivalence classes of sequences seen as observations made by a global sequential observer; the ordering of independent actions in such an observation can be seen as subjective or irrelevant, *i. e.* observations *wabw'* and *wbaw'* are equivalent for independent actions a, b. On the other hand, traces correspond to tuples of local sequential observations: Dealing with distributed systems it is very natural to assume that we have local observers only; provided their sequential observations are consistent in some way, we can reconstruct from these observations a unique partial order of action occurrences.

Furthermore, traces have a very suitable representation as so called dependence graphs.

A basic assumption in trace theory is that dependent actions cannot occur simultaneously. In this note we will drop this assumption: It may very well be possible that actions influencing the same object can happen together and in fact their effect on that object may differ from the effect they have if they occur in some order – think of pressing the control key and some other key on your computer keyboard. Another example is a multiple assignment like $(x, y) := (y, x)$, which exchanges the values of x and y; executing $x := y$ and $y := x$ in some order does not have this effect. (Independence of assignments is studied e. g. in [2].)

Even if the two effects are the same, it is natural to assume that a local observer can observe the simultaneity of two dependent actions, e. g. if two processes read the same file simultaneously this is observed by the file manager. Also other researchers have felt the need to express the actual simultaneity of actions, e. g. for event structures this is done in [10]. A

corresponding description for executions of place/transition nets can be found in [9].

In the approach taken here dependent actions must always be ordered, but now this includes the possibility that they are simultaneous, *i.e.* form a multiset or step. Hence the basic building blocks of our step traces are compound actions, each consisting of several simultaneous actions. Independent actions are not directly ordered, but analogously to the above we have: If independent actions a, b are both dependent to c, and both a and b occur simultaneously with c, then a and b occur simultaneously. Thus the compound actions may also contain independent actions, they are so-called connected steps.

In our approach observations are step sequences: An observer may observe the simultaneity of actions, but otherwise an observation is sequential. Then we can show that step traces correspond to sets of global observations and they correspond to tuples of local observations. Furthermore we can also generalize the concept of dependence graphs to step traces.

Dependence graphs and global observations are dealt with in Section 2 and 3. Their treatment is more or less standard trace theory applied to the alphabet of connected steps. Section 4 deals with local observations; their treatment does not have a counterpart in the standard theory, since here the connected steps get decomposed, while a decomposition of actions is not possible in standard trace theory.

2. STEP SEQUENCES, STEP TRACES AND DEPENDENCE GRAPHS

Following [7] we define a *concurrent system* as a finite bipartite undirected graph (A, X, K), henceforth fixed, with vertex classes A (actions) and X (objects) where $ax \in K$ means that a *accesses* x. We assume that every action needs some object, *i.e.* no vertex of A is isolated. $\mathcal{M}(A)$ is the set of *steps*, of non-empty multisets over A. A step consists of actions observed simultaneously. Formally steps are functions, thus we can e.g. add them, but we also use set-theoretic notation like $a \in s$ for an action a and a step s. We say that a step s *accesses* some object $x \in X$ if more some $a \in s$ we have $ax \in K$. $\mathcal{M}(A)^*$ is the set of *step sequences*. The *graph of a step* s is the subgraph of (A, X, K) induced by $\{a \mid a \in s\} \cup \{x \mid \exists a \in s : ax \in K\}$. A step is called *connected* if its graph is connected, $\mathcal{C}$ is the set of connected steps.

Two actions accesing the same object and appearing to be simultaneous for one observer will also appear to be simultaneous for every other observer, thus they are objectively simultaneous. Hence the actions of a connected step

s are objectively simultaneous. Hence the actions of a connected step s are objectively simultaneous: If $a_1, a_n \in s$ then there is a path $a_1 x_1 a_2 x_2 \ldots x_{n-1} a_n$ in the graph of s, thus a_i and a_{i+1} are objectively simultaneous and by transitivity a_1 and a_n are simultaneous, too. Since no action is isolated this also covers multiple occurrences of one action.

Steps s and s' are *dependent*, $s \mathrm{D} s'$, if there are $a \in s$, $a' \in s'$, $x \in \mathrm{X}$ such that $ax, a'x \in \mathrm{K}$, *independent* otherwise, $x \mathrm{I} s'$. For independent steps their order is subjective or irrelevant.

A connected step s' is a *connected component* of a step s if there is a step s'' such that $s' \mathrm{I} s''$ and $s = s' + s''$. In this situation the graph of s' is a connected component of the graph of s.

To describe an execution we give the order of action occurrences, where the possibilities are 'objectively before/after', 'objectively simultaneous', 'not objectively comparable'. The before-relation is a partial order, and we express objective simultaneity by labelling the elements of this partial order by connected steps. (Another possibility to express simultaneity would be to use pre-orders where objectively simultaneous actions would be ordered both ways). If two elements of this partial order are unordered, then the actions in their labels are simultaneous for some, but not for all observers, *i.e.* they are subjectively simultaneous.

Basic for this description is the assumption: If a happens before b and b together with c, then a happens before c. This assumption is very natural, at least if we can view the actions as being instantaneous.

DEFINITION: *A step trace is a labelled partial order* $(E, \leqq, l)$ *where* E *is a finite set (of events or action occurrences),* $(E, \leqq)$ *is a partial order, and* $l: E \to \mathscr{C}$ *is a function called labelling, such that for all* $e, e' \in E$:

(i) $l(e) \mathrm{D} l(e') \Rightarrow e \leqq e' \vee e' \leqq e$ (*i.e. e, e' are comparable*).

(ii) *If e is an immediate predecessor of e', then* $l(e) \mathrm{D} l(e')$.

We will distinguish step traces and other graph or order theoretic objects only up to isomorphism.

A step trace where all labels belong to A (which can be seen as a subset of $\mathscr{C}$) is a trace in the usual sense.

Next we will generalize dependence graphs to our setting of step traces and show their close relationship to step traces.

DEFINITION: *A dependence graph* (E, F, l) *consists of an acyclic directed graph* (E, F) *and a labelling function* $l: E \to \mathscr{C}$ *such that*

$$\forall e, e' \in E: \quad ee' \in F \vee e'e \in F \vee e = e' \Leftrightarrow l(e) \mathrm{D} l(e').$$

PROPOSITION 2.1: *For a step trace* $t=(E, \leqq, l)$ *define* $dep(t)=(E, F, l)$ *by* $ee' \in F \Leftrightarrow e<e' \wedge l(e)\, Dl(e')$. *Then dep is a bijection from step traces onto dependence graphs.*

Proof: First observe that *dep* is well-defined, *i.e.* if $(E, \leqq, l)$ and $(E', \leqq', l')$ are isomorphic then (E, F, l) and (E', F', l') defined as above are isomorphic, too. Furthermore $dep(t)$ is indeed a dependence graph.

For a dependence graph (E, F, l) define $st(E, F, l)=(E, \leqq, l)$ by: $e<e'$ if and only if there exists a directed path in (E, F, l) form e to e'. It is easily checked that *st* is well-defined, and that $st(E, F, l)$ is a step trace.

Now the claim follows since $dep \circ st$ and $st \circ dep$ are the respective identity functions. □

For a step trace $t=(E, \leqq, l)$ and a set $E' \subseteqq E$ of maximal elements let $t-\{l(e) \mid e \in E'\}$ be the labelled partial suborder of t induced by $E-E'$. For a dependence graph $g=(E, F, l)$ and a set $E' \subseteqq E$ of vertices without outgoing edges let $g-\{l(e) \mid e \in E'\}$ be the labelled subgraph of g induced by $E-E'$.

LEMMA 2.2: (i) *Let t be a step trace,* $C \subseteqq \mathscr{C}$. *Then* $t-C$ *is defined if and only if* $dep(t)-C$ *is defined. In that case,* $t-C$ *and* $dep(t)-C$ *are well defined, they are a step trace, a dependence graph resp., and* $dep(t-C)=dep(t)-C$.

(ii) *Let* t, t' *be step traces,* $C \subseteqq \mathscr{C}$ *such that* $t-C$, $t'-C$ *are defined. Then* $t-C=t'-C$ *implies* $t=t'$.

Proof: (i) Since maximal elements of t correspond to vertices of $dep(t)$ without outgoing edges, the first claim is immediate. Maximal elements of t are incomparable, hence their labels are independent, and especially they are different. Hence maximal elements can uniquely be identified by their labels. With an analogous consideration for dependence graphs well-definedness follows. The rest is immediate from the definitions.

(ii) Given $dep(t-C)=(E, F, l)$ and C, we can uniquely construct $dep(t)=(E \dot{\cup} C, F', l')$ by $F'=F \cup \{ec \mid e \in E, c \in C, l(e)\, D\, c\}$, $l'(e)=l(e)$ for $e \in E$ and $l(c)=c$ for $c \in C$. Hence the result follows. □

From a step trace t one can read off a corresponding step sequence as follows: If C is a set of labels of maximal elements of t, then concatenate a step sequence corresponding to $t-C$ with that step which is the sum of the labels in C, *i.e.* the elements of t corresponding to C, which are pairwise incomparable, form together the last step of this sequence. In the next section we will see that a step sequence corresponding to a step trace t can be seen as a global observation of the system execution t. Formally:

DEFINITION: *For step traces t we define $Step(t)$, the set of step sequences corresponding to t by:*

(i) $Step(\varnothing, \varnothing, \varnothing) = \{\lambda\}$, *where λ denotes the empty sequence.*

(ii) *If* $t \neq (\varnothing, \varnothing, \varnothing)$, *then* $Step(t) = \bigcup_C Step(t-C).(\sum_{c \in C} c)$, *where C ranges over the non-empty subsets of $\mathscr{C}$ such that $t-C$ is defined.*

The next theorem shows that the sets $Step(t)$ form a partition of $\mathscr{M}(A)^*$:

THEOREM 2.3: (*i*) *If t is a step trace, then* $Step(t) \neq 0$.

(ii) *For each step sequence w there is a unique step trace $t(w)$ such that* $w \in Step(t(w))$.

Proof: (i) Obvious from the definition and Lemma 2.2.

(ii) Let $w = w_1 \ldots w_n$, $w_i \in \mathscr{M}(A)$. We define $t(w)$ by defining

$$dep(t(w)) = (E, F, l)$$

with

$$E = \{(c, i) \mid c \text{ is a connected component of } w_i\},$$
$$(c, i)(c', j) \in F \Leftrightarrow i < j \wedge c\, D\, c',$$
$$l(c, i) = c.$$

Obviously, the connected components of w_n correspond to maximal elements of $t(w)$, hence one easily sees by induction on n that $w \in Step(t(w))$. Vice versa, if $w \in Step(t)$ for some trace t, then w_n is the sum of $l(e)$, $e \in E'$, for some set E' of maximal elements of t. By definition, these $l(e)$ are connected steps and they are pairwise independent, since the elements of E' are pairwise incomparable. Thus the $l(e)$, $e \in E'$, are connected components of w_n, and we get a label preserving bijection from E' to $\{(c, i) \in E \mid i = n\}$. Thus for $C = \{l(e) \mid e \in E'\}$ we get $t - C = t(w_1 \ldots w_{n-1})$ by induction, hence $t - C = t(w) - C$ and $t = t(w)$ by Lemma 2.2. □

We will now define a concatenation for step traces (in the usual way) which is based on a concatenation of the corresponding dependence graphs, and we will observe that this concatenation corresponds to the concatenation of step sequences.

DEFINITION: *Let* (E_1, F_1, l_1), (E_2, F_2, l_2) *be dependence graphs with* $E_1 \cap E_2 = \varnothing$. *Then* $(E, F, l) = (E_1, F_1, l_1).(E_2, F_2, l_2)$ *is defined by*

$$E = E_1 \cup E_2.$$
$$F = F_1 \cup F_2 \cup \{e_1 e_2 \mid e_1 \in E_1, e_2 \in E_2, l_1(e_1)\, D\, l_2(e_2)\}$$
$$l = l_1 \cup l_2.$$

For step traces t, t' the step trace $t.t'$ is the step trace corresponding to $dep(t).dep(t')$.

It is easy to see that (E, F, l) as defined above is a dependence graph indeed, hece $t.t'$ is really defined.

THEOREM 2.4: *For step sequences w, w' we have $t(ww') = t(w).t(w')$.*

Proof: Obvious from the construction in the proof of Theorem 2.3. □

This result shows that step traces, dependence graphs and equivalence classes $Step(t)$ of step sequences form isomorphic monoids with respect to concatenation.

3. GLOBAL OBSERVATIONS

As discussed in the introduction, step sequences can be seen as global observations where we regard the ordering of independent steps as irrelevant or subjective. In other words, for independent steps s, s' the step $s+s'$ and the sequences ss' and $s's$ are different observations of the same execution of the system. Therefore we define the following congruence, where a congruence class consists of all possible global observations of one execution of the system.

DEFINITION: *Let $\equiv$ be the least congruence (w.r.t. concatenation) on $\mathcal{M}(A)^*$ such that for all independent steps s, s' we have $s+s' \equiv ss'$.*

Usually the basic congruence in trace theory is of the form $aa' \equiv a'a$ for independent actions a, a'. Since actions are special steps this congruence follows from our definition ($aa' \equiv a+a' \equiv a'a$).

THEOREM 3.1: *For step sequences w, w' we have $w \equiv w'$ if and only if $t(w) = t(w')$.*

Proof: "$\Rightarrow$" If s, s' are independent steps, then the set of connected components of $s+s'$ is the disjoint union of the sets of connected components of s and s'. Thus if we replace in a step sequence w the step $s+s'$ by ss' or vice versa the construction in the proof of Theorem 2.3 yields the same result for both sequences (up to isomorphism).

"$\Leftarrow$" By induction on the number of connected components, where the claim is obvious for zero components. Otherwise we have maximal elements e, e' in $t(w)$, $t(w')$ with the same label c. By the construction of Theorem 2.3 c corresponds to a connected component of some step in w (w') that is

independent of all steps appearing after it in w (w'). Thus $w \equiv w_1 c$, $w' \equiv w_1' c$, $t(w_1) = t(w) - \{c\}$ and $t(w_1') = t(w') - \{c\}$. By induction we have $w_1 \equiv w_1'$, therefore $w \equiv w'$. □

Since by this result the congruence classes of global observations defined in this section are just the sets $Step(t)$ for step traces t, we have already seen that the classes of global observations form a monoid isomorphic to the step trace monoid.

4. LOCAL OBSERVATIONS

In this section we assume that instead of some global observation we only have a tuple of local observations. Each local observation corresponds to one object and includes only those actions that access this object.

It is here that we need the representation of a concurrent system as a bipartite graph containing actions and objects. It is more usual to define the system as (A, D), where D is the dependence relation (for actions only) we have defined above. Up to now we could have just as well worked with such a representation. Each object x defines a clique of (A, D), namely the set $\{a \in A \mid ax \in K\}$, and these cliques cover the graph (A, D), in the sense that each vertex $a \in A$ and each edge $ab \in D$ is contained in some clique. Therefore the reader may also think of a concurrent system as a graph (A, D) given together with a family X of cliques which cover the graph. With this view, Theorem 4.3 below generalizes the well-known embedding theorem, which states that every trace monoid can be embedded in a direct product of free monoids, see [4, 5].

Of course, local observations of the same execution have to be consistent; e. g. if one object 'observed' two occurrences of a, some other observed only one occurrence of a, then these observations cannot belong to the same execution of the system. We regard local observations as consistent if they can be seen as suitable restrictions of the same step sequence. We will deal with consistent local observations only:

DEFINITION: *For $w \in \mathcal{M}(A)^*$ and $x \in X$ the local observation $loc(w, x)$ of w by x is defined inductively:*

(i) $loc(\lambda, x) = \lambda$

(ii) *Let s' be the restriction of $s \in \mathcal{M}(A)$ to $\{a \mid ax \in K\}$. Then*

$$loc(ws, x) = \begin{cases} loc(w, x) & \text{if } s' = \varnothing \\ loc(w, x)\, s' & \text{if } s' \neq \varnothing \end{cases}$$

Each step of $loc(w, x)$ comes from a connected component of some step of w, namely that component which accesses x; these connected components are dependent and therefore totally ordered in the step trace corresponding to w. Thus we can read off $loc(w, x)$ from $t(w)$ as well, that is we get the following definition and lemma.

DEFINITION: *Let t be a step trace, $x \in X$. The elements of t whose labels access x are totally ordered, let w be the sequence of their labels. Then define $loc(t, x) = loc(w, x)$.*

LEMMA 4.1: *For all step sequences w and $x \in X$: $loc(w, x) = loc(t(w), x)$.*

After the next lemma we are ready to prove that a step trace corresponds to those step sequences that give rise to the same local observations.

LEMMA 4.2: *Let $c \in \mathscr{C}$ and $t = (E, \leqq, l)$ be a step trace. Then there is a maximal element e of t with $l(e) = c$ if and only if for all $x \in X$ that are accessed by c the last step of $loc(t, x)$ is c restricted to $\{a \mid ax \in K\}$.*

Proof: "$\Rightarrow$" obvious.

"$\Leftarrow$" Since c is non-empty there exists some $x \in X$ which is accessed by c. Hence by assumption there exist elements of E whose labels access a common object with c. Let e be maximal among these elements, and let x be accessed by $l(e)$ and c.

First we show that $l(e) = c$. Assume to the contrary. Then there exists $a \in l(e)$ or $a \in c$ with $l(e)(a) \neq c(a)$ (remember that steps are functions!). In the first case consider a path in the graph of $l(e)$, which is connected, from e to a. On this path we can find some $x' \in X$ that is accessed by $l(e)$ and c and some $a' \in A$ that accesses x' such that $l(e)(a') \neq c(a')$. Analogously we find such x' and a' in the second case by considering a path in the graph of c. By definition of $loc(t, x')$ and the choice of e we conclude that the last step of $loc(t, x')$ is a restriction of $l(e)$, by assumption it is a restriction of c, thus $l(e)(a') = c(a')$, a contradiction.

Now it is easy to see that e is maximal in t: Otherwise we would find an immediate successor e' of e and by definition of a step trace $l(e')$ and $l(e)$ access a common object, a contradiction. □

THEOREM 4.3: *For all step sequences w and w':*

$$t(w) = t(w') \quad \Leftrightarrow \quad \text{for all } x \in X\text{: } loc(w, x) = loc(w', x).$$

Proof: "$\Rightarrow$" Lemma 4.1.

"$\Leftarrow$" By induction on the number of connected components, the claim being obvious for $w = \lambda$. Choose a maximal e in $t(w)$. By two applications of

Lemma 4.2 we can find a maximal e' in $t(w')$ with $l(e)=l'(e')$. By induction $t(w)-\{l(e)\}=t(w')-\{l'(e')\}$ and Lemma 2.2 (ii) yields the result. □

This result can also be seen as saying that a suitable synchronization operator applied to the local observations yields the step trace. This synchronization does not only identify occurrences of the same action in different observations, it also merges overlapping steps to larger steps.

We conclude by remarking that obviously we have for step sequences w and w' and $x\in X$ that $loc(ww', x)=loc(w, x)\,loc(w', x)$. Hence we can define a componentwise concatenation on tuples $(loc(w, x))_{x\in X}$ which corresponds to the concatenation of step sequences. Therefore Theorem 4.3 does not only exhibit a bijection between step traces and tuples of local observations, it also follows immediately that step traces and tuples of local observations form isomorphic monoids.

ACKNOWLEDGEMENTS

I thank Prof. Brauer, V. Diekert, and R. Gold for their valuable comments, and H. Hadwiger for the excellent typing.

REFERENCES

1. I. J. Aalbersberg and G. Rozenberg, Theory of Traces, *Theor. Comp. Sci.*, 1988, 60, pp. 1-82.
2. E. Best and C. Lengauer, Semantic Independence, *Sci. Comput. Prog.*, 1989, 13, pp. 23-50.
3. P. Cartier and D. Foata, Problèmes Combinatoires de Commutation et Réarrengements, *Lect. Notes in Math.*, No. 85, Springer, 1969.
4. M. Clerbout and M. Latteux, Partial Commutations and Faithful rational Transductions, *Theor. Comp. Sci.*, 1985, 35, pp. 241-254.
5. R. Cori and D. Perrin, Automates et Commutations partielles, *R.A.I.R.O.-Informatique théorique et Application*, 1985, 19, pp. 21-32.
6. G. Lallement, *Semigroups and Combinatorial Applications*, John Wiley and Sons, New York, 1979.
7. A. Mazurkiewicz, Traces, Histories, Graphs: Instances of a Process Monoid, *in* M. P. Chytil *et. al.* Ed., *Proceeding of the 11th Symposium on Mathematical Foundations of Computer Science* (MFCS); *Lect. Notes Comp. Sci.*, 1984, 176, pp. 115-133.
8. A. Mazurkiewicz, Trace Theory, *in* W. Brauer *et. al.* Ed., *Petri Nets: Applications and Relationships to other Models of Concurrency; Lect. Notes Comp. Sci.*, 1987, 255, pp. 279-324.
9. W. Vogler, Executions of Petri Nets, in *Proc. of 8th European Workshop on Application and Theory of Petri Nets, Zaragoza*, 1987, pp. 551-564. Also in: Tech. report TUM-I8806, Techn. Univ. München, 1988.
10. J. Winkowski, An Equivalence of Communicating Processes in Distributed Environments, *Fundamenta Informaticae*, 1989, *XII*, pp. 97-128.