The Difference between Splitting in n and $n+1$

(abstract)

Rob van Glabbeek
Computer Science Department
Stanford University
Stanford, California 94305, USA
rvg@cs.stanford.edu

Frits Vaandrager
MIT Laboratory for Computer Science
545 Technology Square
Cambridge, MA 02139, USA
frits@theory.lcs.mit.edu

February 18, 1991

It is established that durational and structural aspects of actions can in general not be modeled in interleaving semantics, even when time-consuming actions are represented by pairs of instantaneous actions denoting their start and finish. By means of a series of counterexamples it is shown that, for any n, it makes a difference whether actions are split in n or in $n+1$ parts.

Note: Work on this paper was initiated at the time both authors were employed at CWI, Amsterdam, continued during a visit of the second author to the Technical University of Munich, at the time the first author was employed there, and finalized while the authors were employed at Stanford resp. MIT. The work of the first author at the TU Munich was supported by Sonderforschungsbereich 342, and the work at Stanford University by the National Science Foundation under grant number CCR-8814921. The work of the second author at MIT was supported by ONR contract N00014-85-K-0168.

In event based models of concurrency which interleave concurrent actions, and thus reduce parallelism to nondeterminism, some rather strong assumptions have to be made about the nature of events. The following formulation of these assumptions is taken from HOARE [8, page 24]:

> *The actual occurrence of each event in the life of an object should be regarded as an instantaneous or an atomic action without duration. Extended or time-consuming actions should be represented by a pair of events, the first denoting its start and the second denoting its finish.*

In interleaving semantics the behavior of a system that performs two actions a and b in parallel is considered the same as the behavior of a system that either does an a followed by a b, or a b followed by an a. Algebraically this leads to equations like

$$a \| b \;=\; a \cdot b + b \cdot a.$$

Such identifications are reasonable under the assumption that the actions a and b are instantaneous, but become highly problematic if a and b are durational or time consuming. Any intuition that in $a||b$ the a and the b are causally independent and can occur simultaneously, or that $a||b$ is faster than $a \cdot b + b \cdot a$, cannot be captured in interleaving semantics in terms of primitive notions. The postulate of interleaving forces one to consider instantaneous actions only. Hoare however, and with him many other advocates of interleaving semantics, argues that this is not really a restriction because one can still deal with durational actions, by *splitting* such actions into a beginning and an end. In interleaving semantics

$$a^+ \cdot a^- \;||\; b^+ \cdot b^- \;\neq\; a^+ \cdot a^- \cdot b^+ \cdot b^- \;+\; b^+ \cdot b^- \cdot a^+ \cdot a^-,$$

because the process on the left has a trace $a^+b^+a^-b^-$ which the process on the right has not. Observations like this make many people believe that at least in theory the idea of splitting offers a viable approach to the concept of durational actions.

In this paper we argue that often splitting, i.e. the use of pairs of instantaneous actions to model single durational actions, is *unnatural* and blurs the useful distinction between issues of functional behavior and performance. Our main result however is technical and of a more serious nature: we show that the representation, in interleaving models, of durational actions by pairs of actions is simply *impossible* in a general setting of concurrent and nondeterministic systems. Basically, the problem is that in the presence of a sufficient amount of nondeterminism and concurrency, it is not always clear how start actions match up with finish actions. This confusion leads to the identification of certain processes that should be distinguished by any semantic equivalence which claims to capture the notion of actions with duration. Our counterexamples that illustrate these phenomena are rather complex and don't leap out at the casual observer. PRATT [13] compared this situation with the 19th century engineers who did not notice discrepancies in their day-to-day work due to relativity and quantum mechanics. And indeed, splitting works in all practical situations that we know of. The point which we try to make is that when applications become more complex, this situation may change because splitting is unable to capture the notion of durational actions in general. On top of this, we argue that, even in those cases in which it does work, splitting of durational actions is often not a natural thing to do. Given the widespread belief in the universal power of splitting, we proceed in a very careful manner, with the aim to eradicate this misconception, in all its possible forms, once and for all. However, we would like to stress in advance that our results by no means imply that in a setting with durational actions the use of partial order or ST-semantics is 'superior' to a semantics based on interleaving of split actions. In fact we think that in many specific areas of application interleaving+splitting can lead to simpler and more satisfactory solutions. A typical example here is the area of wait-free atomic registers, where apparently the use of interleaving based models in combination with splitting of the durational read and write operations (see for instance [3]), is more successful than the use of the partial order based model of [9].

As it is more general and much simpler, the discussion of this paper will take place at the level of semantic models and not at the level of the languages which are interpreted in those models. We use semantic models which are based on partial orders of events, because in these models we can easily express that actions have duration or structure. In order to argue that a combination of interleaving and splitting cannot capture these features, we need a model which can! A simple way to capture the notion of durational actions in a partial order model is to introduce a mapping which associates a duration to each action. Given an execution sequence, one can construct a *real-time execution sequence* by labeling each event with a start time and a finish time, in such a way that (1) the starting times of consecutive events in the sequence are nondecreasing, (2) the start time of each event is greater or equal than the finish time of all its predecessors in the partial order, and (3) for each individual event the difference between finish and start time equals the duration of the action which is associated to the event. Partial order models can also capture the concept of actions having structure, via *action refinement*: the operation of refining actions into more complicated processes. Refinement of actions is advocated in [12, 4, 6] as a natural operation on processes, allowing to decrease the level of abstraction at which systems are described. Splitting of actions can be viewed as a simple form of action refinement in which each action a is refined into a process which first does a^+ and then a^-. Given the complexity of our counterexamples, we decided to split the technical part of this paper into two parts. First we discuss the issue in the linear time model of *pomset processes*, and next we discuss it again in the branching time model of *labeled prime event structures*. The results of the second part are more general, but also more difficult to understand because the counterexamples have to preserve branching structure too. Each counterexample consists of two systems (represented either as pomset processes or as event structures) which are clearly different if we assume that actions have duration, but which are identified if we first split the events in each system and then compare the interleaving behaviors of the resulting systems.

The first counterexample we found (the *owl example*) consists of two such systems which are identified in interleaving semantics after splitting each action into two parts, but which can be distinguished by splitting each action into three parts. This may lead a fervent splitter to speculate that although durational and structural properties of actions can not be modeled in interleaving semantics by simply splitting actions into a beginning and an end, it might still be possible to do so by splitting actions in three or more parts. Therefore we also consider n-splitting, where a sequence $a_1 \cdot \ldots \cdot a_n$ is substituted for each action a, and show that for any $n \geq 2$ there are processes with the same interleaving behavior after splitting each action in n parts, but with different interleaving behavior after each action is split into $n+1$ parts. These examples show that it is impossible to capture durational or structural properties of actions by splitting them into any finite number of parts.

Even though we will not discuss this issue in any detail, we claim that our counterexamples can be recreated on the language level in all major process algebras. In the language CCSP of [10], for instance, any finite prime event structure can be denoted up to history preserving bisimulation equivalence. Since all our counterexamples involve finite structures only, this observation immediately implies that they can be translated to CCSP. Aceto & Hennessy [2] discuss in detail the translation of one of our counterexamples into Milner's CCS. All counterexamples necessarily involve *autoconcurrency*, the possibility that two instances of the same action occur simultaneously. If there is no autoconcurrency, then there can be no confusion about how finish actions match up with start actions, and splitting becomes a consistent way of dealing with durational actions. The subset of the language *Rec* of [11] which contains injective morphisms only, is a nontrivial example of a language without autoconcurrency. Although we do not exclude that such a restricted language can be useful in practice, our feeling is that autoconcurrency, just like nondeterminism, is a feature which arises naturally if one tries to give abstract descriptions of concurrent systems. Therefore we think that a theory of durational actions will be more useful if it allows for autoconcurrency.

For each interleaving equivalence one can define a *split-n* equivalence, identifying two processes exactly when they are interleaving equivalent after splitting each action into n parts. Our counterexamples show that the split-n equivalences are too course to capture durational and structural aspects of actions. This means that the interleaving equivalences are too course for this purpose even if occurrences of actions are represented by a sequence of two or more events. In a previous paper [7] we introduced a type of equivalence that is very similar to split-2 equivalence, but which in addition requires that the way start actions match up with finish actions is the same for equivalent processes. These so-called *ST-equivalences* are finer than the split-n equivalences but courser then the main partial order equivalences. In [7, 5] is was established that they do not suffer from the problems we here encounter with split equivalences, and are as suitable for modeling durational actions as are the main partial order equivalences. In ST-semantics as well as in partial order semantics no additional discriminatory power is obtained when actions are split into a beginning and an end (or in more than two parts) before their behaviors are compared. So whereas in interleaving semantics splitting is unnatural and insufficient, in ST- and in partial order semantics it is unnatural and unnecessary. In order to capture durational and structural properties of actions one needs to make the distinctions of ST-semantics. The additional distinctions made by partial order equivalences do not harm, but are inessential for this purpose.

ACETO & HENNESSY [1] investigated the largest congruence contained in (interleaved) bisimulation equivalence for a simple language without communication and restriction (a subset of CCS), but with a general refinement operator. They characterized this congruence as the equivalence which declares two processes equal iff after splitting actions into begin and end, the results are (interleaved) bisimilar. It follows that our counterexamples do not carry over to the language considered by Aceto and Hennessy, and that their result does not carry over to full CCS with refinement. In fact, on the domain of processes that can be expressed in Aceto and Hennessy's language, split equivalence, ST-equivalence and partial order equivalence all coincide.

References

[1] L. Aceto and M. Hennessy. Towards action-refinement in process algebras. In *Proceedings 4^{th} Annual Symposium on Logic in Computer Science,* Asilomar, California, pages 138–145. IEEE Computer Society Press, 1989. Full version to appear in *Information and Computation.*

[2] L. Aceto and M. Hennessy. Adding action refinement to a finite process algebra. Report 6/90, Computer Science, University of Sussex, Brighton, 1990.

[3] B. Bloom. Constructing two-writer atomic registers. In *Proceedings of the 6^{th} Annual ACM Symposium on Principles of Distributed Computing,* Vancouver, Canada, pages 249–259, 1987. Also to appear in special issue *IEEE Transactions on Computers.*

[4] L. Castellano, G. De Michelis, and L. Pomello. Concurrency vs interleaving: an instructive example. *Bulletin of the European Association for Theoretical Computer Science*, 31:12–15, 1987.

[5] R.J. van Glabbeek. The refinement theorem for ST-bisimulation semantics. In M. Broy and C.B. Jones, editors, *Proceedings IFIP Working Conference on Programming Concepts and Methods,* Sea of Gallilea, Israel. North-Holland, 1990.

[6] R.J. van Glabbeek and U. Goltz. Refinement of actions in causality based models. In J.W. de Bakker, W.-P. de Roever, and G. Rozenberg, editors, *REX Workshop on Stepwise Refinement of Distributed Systems: Models, Formalism, Correctness,* Mook, The Netherlands 1989, volume 430 of *Lecture Notes in Computer Science*, pages 267–300. Springer-Verlag, 1990.

[7] R.J. van Glabbeek and F.W. Vaandrager. Petri net models for algebraic theories of concurrency. In J.W. de Bakker, A.J. Nijman, and P.C. Treleaven, editors, *Proceedings PARLE conference,* Eindhoven, *Vol. II (Parallel Languages)*, volume 259 of *Lecture Notes in Computer Science*, pages 224–242. Springer-Verlag, 1987.

[8] C.A.R. Hoare. *Communicating Sequential Processes.* Prentice-Hall International, Englewood Cliffs, 1985.

[9] L. Lamport. On interprocess communication. *Distributed Computing*, 1:77–101, 1986.

[10] E.-R. Olderog. Operational Petri net semantics for CCSP. In G. Rozenberg, editor, *Advances in Petri Nets 1987*, volume 266 of *Lecture Notes in Computer Science*, pages 196–223. Springer-Verlag, 1987.

[11] E.-R. Olderog. *Nets, Terms and Formulas: Three Views of Concurrent Processes and Their Relationship.* Hab., Institut für Informatik und Praktische Matematik, Christian-Albrechts-Universität zu Kiel, Fed. Rep. Germany, 1988.

[12] V.R. Pratt. Modelling concurrency with partial orders. *International Journal of Parallel Programming*, 15(1):33–71, 1986.

[13] V.R. Pratt. DO the great debate CONTINUE. Message to mailing list concurrency@theory.lcs.mit.edu, 13 Nov 1990.