Math. Struct. in Comp. Science (2003), *vol.* 13, *pp.* 481–484.
DOI: 10.1017/S096012950300402X Printed in the United Kingdom

Foreword to special issue: The difference between concurrent and sequential computation

Including selected papers from: EXPRESS'00 – 7th International Workshop on Expressiveness in Concurrency

LUCA ACETO[†], GIUSEPPE LONGO[‡] and BJÖRN VICTOR[§]

[†]*Aalborg, Denmark*
[‡]*Paris, France*
[§]*Uppsala, Sweden*

Received 20 April 2002

Computer Science has witnessed the emergence of a plethora of different logics, models and paradigms for the description of computation. Yet, the classic Church–Turing thesis may be seen as indicating that all general models of computation are equivalent. Alan Perlis referred to this as the 'Turing tarpit', and argued that some of the most crucial distinctions in computing methodology, such as sequential *versus* parallel, deterministic *versus* non-deterministic, local *versus* distributed disappear if all one sees in computation is pure symbol pushing. How can we express formally the difference between these models of computation?

This double issue of *Mathematical Structures in Computer Science* aims to address this fundamental question by focussing on the dichotomy between sequential and concurrent computation. In particular, on ways to capture formally that any translation from an expressive approach to concurrency to a formalism like that of Turing Machines would miss 'what really matters'. By way of an example, physicists know very well that non-Euclidean geometries can be 'embedded' into the Euclidean one (and conversely), but they also know that the equivalence, an interesting equicoherence, misses what really matters, for example, the intrinsic geometry of physical (typically relativistic) space. Similarly, one can give an 'isomorphism' (as sets) between the real line and the plane (or any finite dimensional space), but this misses all that matters in the mathematical understanding of space, for example, the notion of neighbourhood (Cartesian dimension is a topological invariant, not a set-theoretic one).

Two questions, in particular, arise:

— Are there theorems or rigorous arguments to confirm the view that translating concurrency into sequential computation misses some important aspect of concurrency?
— What is the peculiar role played by space (and time) in distributed, concurrent systems?

We solicited papers addressing these questions, and related issues, in a creative and novel way. In addition, since we thought that the general theme of the workshop 'EXPRESS'00: 7th International Workshop on Expressiveness in Concurrency' dealt with problems that are closely related to those raised above, we invited authors of selected papers presented at the workshop to submit full versions of their contributions to this special issue.

The nine papers that appear in this double issue of *Mathematical Structures in Computer Science* have been selected amongst the submissions that we received after a strict refereeing process, and address the questions we raised above from a variety of angles.

The first five papers appear in this first part of the double issue:

— A popular viewpoint in addressing the differences between sequential and concurrent computation draws upon the following temporal distinction.

Sequential behaviour allows only one event to happen at a time. With concurrent behaviour multiple events may occur simultaneously.

In his contribution, *Pratt* suggests replacing this time-centric viewpoint with one based more symmetrically on time and information as complementary or dual primitives. He then extends this proposal by refining the simplest non-trivial view of atomic time, *viz.* the boolean view that an event has or has not happened, to take into account the fact that, in the presence of concurrency and conflict, events may be in an intermediate state of *transition* or in a state of *cancellation*. This results in a novel, logical view of Winskel's model of event structures.

— The aim of *Abramsky*'s contribution is to use games and logic to understand some facets of the distinction between sequentiality and concurrency. He shows that the simple, intuitive notion of polarized games due to Blass leads to a surprising problem of non-associativity of composition, which can be traced to an incompatibility between an interleaving view of computation and logic in a classical format. Two ways out of this dilemma are described: one is to keep sequentiality but to restrict the logic by the use of polarities (as in Laurent's recent system of Polarized Linear Logic) or focussing (as introduced by Andreoli and Pareschi, and used in Girard's Ludics). The other is to use 'truly concurrent games' to resolve the problem of associativity while still keeping the full classical system.

— Another view of the distinction between sequentiality and concurrency is offered by the theories of computability and complexity. It is by now well known that bisimulation equivalence over several classes of infinite state systems is decidable, whereas language equivalence is not. It is, however, an open problem whether weak bisimilarity is decidable for Basic Process Algebra and Basic Parallel Processes. In his contribution, *Srba* offers a PSPACE-hardness result for the complexity of weak bisimulation over Basic Parallel Processes, and demonstrates DP-hardness for the problem of deciding weak regularity of processes in Basic Process Algebra.

— In developing models of computation, we are often faced with the desire to extend our modelling abilities to faithfully account for some aspects of physical reality. In particular, with the increasing deployment of embedded software, modelling of real-time issues in computation has become more and more important. The contributions of *Baeten* and *Corradini and Di Cola* address the addition of real-time to process algebraic

specification languages for concurrent computation from two different perspectives. Baeten presents a re-design of process algebraic languages with or without timing with the aim of obtaining a more uniform and modular theory. Corradini and Di Cola show how the expressive power of a language for the description of timed processes strongly affects the discriminating power of urgent and patient actions.

The remaining papers will appear in the second part of this double issue. The first three offer contributions related to the expressiveness of different languages for describing mobile computing – which is one of the computing paradigms in which one is essentially forced to have some representation of (logical) distribution in computation.

— The contribution by *Laneve and Victor* is devoted to a study of the expressiveness of a calculus of mobile processes without prefix or summation, called the *solos calculus*. Using two different encodings, they show that the solos calculus can express both action prefix and guarded summation, even when actions carry at most two names. On the other hand, expressiveness is lost in the solos calculus without match and with actions carrying at most one name.
— In one of the most memorable results in the literature on the π-calculus, *Palamidessi* argues that the asynchronous π-calculus, proposed by Honda and Tokoro and, independently, by Boudol, is not as expressive as the full π-calculus. This is achieved by showing that there does not exist any uniform, fully distributed translation from the π-calculus into the asynchronous π-calculus, up to any 'reasonable' notion of equivalence.
— The contribution by *Zimmer* considers the *Pure Safe Ambient Calculus*, which is Levi and Sangiorgi's *Safe Ambient Calculus* (a variant of Cardelli and Gordon's *Mobile Ambient Calculus*) restricted to its mobility primitives, and focusses on its expressive power. The main contribution of this study is an encoding of the synchronous π-calculus into pure ambients, showing that pure ambients are as expressive as the π-calculus.
— Finally, in their contribution, *Matherat and Jaekel* argue that one needs to have realistic models of both the computational structure and the physical implementation of the logical operators that are used to build computing machines in order for these machines to operate correctly, and for the users to be confident in their output. They address the relation between concrete computing machines and physical time, compare the solutions provided by synchronous and asynchronous approaches to the implementation of logical operators, and argue that they reflect different causal structures for space-time.

We trust that the above papers will further stimulate the search for creative and novel ways of addressing the issues we have raised by proposing and editing this special issue of *Mathematical Structures in Computer Science*. In particular, we strongly believe that these contributions highlight the need for suitable measures of the expressive power of languages for the description of computation, which may be used to shed light on the differences between different computing paradigms.

Last, but not least, we express our gratitude to the colleagues who have spent a considerable amount of their time refereeing the submitted papers with a level of commitment to the cause that has been beyond the call of duty. Good editorial work is helped, if not made, by good referees, and it has been a great pleasure to see the very high scientific standards of refereeing that were commonplace for this special issue. Reading the reports of our referees was often a humbling experience, and an example of what it means to be a conscientious member of the scientific community.