EDITOR'S FOREWORD

This Issue of Fundamenta Informaticae consists of four papers on Concurrency, which I shall refer to below as M, W, RT, and BC (the initials of the authors). All of them deal with causal (partial order) semantics, and except for W are based on lectures delivered at the GMD-Workshop, "Combining Compositionality with Concurrency", Königswinter, March 1988.

The papers may be classified according to the following criteria:

First — the view on causality. This is in fact the dilemma between linear and branching time. Whereas M and BC consider the simpler "linear causality", W and RT deal with the more discriminating "branching causality".

Second — the choice of concurrent system. M, W, and RT are concerned with Net models of concurrency, while BC deals with the language CCS.

Significant parts of M and RT are dedicated to the conceptual background and, mainly to synchronization of processes as the main tool for constructing complex concurrent systems from simpler ones. Since M deals only with causality (no explicit involvement of branching), processes are formalized as Qualified Pomset Processes. On the other hand RT focuses on Behavior Structures (BS) as the notion which captures, in one entity, both causality and branching. Accordingly, RT covers the topics of BS-bisimulation, Synchronization of BS and embeddings into BS.

As mentioned above, the papers differ on whether "linear causality" or "branching causality" is intended. In particular for M and W this is the only relevant difference, otherwise they deal with the same topic and contain similar results. Note, however, that for the authors of this issue, such differences by no means characterize their specific views on the controversy "linear time vs branching time". In fact, in additon to M and BC, the authors of these papers have investigated elsewhere topics in the branching-causal setting as well.

Languages such as CCS, TCSP, etc. are based on appropriate repertoires of operations which suggest a compositional way of assembling complex systems from elementary ones. But, Nets (Petri Nets are the paradigm) seem to lack compositional structure. Hence, one might expect compositional (denotational) definitions of semantics for languages and operational ones for Nets.

In fact, as far as interleaving semantics is concerned, in both cases operational semantics was first established and became generally accepted. Namely, for terms — through rules for the derivation of Labeled Transition Systems (LTS) and Petri Nets through firing rules for the token game. But, for causal semantics it indeed happened that the first definitions were in operational style for nets (Petri) and in a compositional style for terms (Winskel, Mycroft, Goltz, Loogen). Recently, however, attention was attracted by attempts to deal in just the opposite way, i.e. to provide compositional semantics for nets and operational semantics for terms.

The compositional approach to semantics of Petri Nets was first formulated by Mazurkievicz. It relies on a novel approach to decomposition of nets and on the fundamental observation that synchronization is a sufficient and adequate tool for composing complex nets from atomic nets. On the other hand, in a series of papers Montanari, Degano and De Nicola proposed an original method to describe causal semantics of CCS-like terms, through operational manipulations with appropriate subterms.

The papers in this issue develop the new insight above in the following ways:

Recall that Mazurkievicz's first work was essentially about a special class of Petri Nets — the so-called C/E Nets. In the more general situation of P/T-Nets one has to reconstruct the causal behavior from a token game in which concurrent multifirings are allowed. This is a relevant generalization which involves new phenomena, that do not occur in Petri's original (C/E) model. M and W provide compositional semantics for P/T Nets. RT achieved the same, independently, using Nets over Processes (and in particular Nets over automata and multi-automata) as a unifying tool for different models of Concurrency, based on the net concept.

BC is inspired by the Montanari-Degano-De Nicola ideas, but elaborates a new approach which closely follows the original way of defining interleaving semantics for CCS via LTS. The novelty is in the use of proved transitions which allow the discerning of causality in situations where it is overlooked by simple transitions.

The importance of having both operational and denotational semantics is well known in Programming Language Theory and the effect is the same in Concurrency Theory; all this under the assumption that both definitions are consistent. After all, the fit between the two definitions is the factor which provides guidance to the design and analysis of Systems. In RT it is shown that this is indeed the case for Nets over automata. Though the present version of BC does not include similar results for the semantics of CCS, the authors actually obtained them (Communication at the GMD-Workshop in Königswinter) and they will be published elsewhere.

Sometimes, slightly different notations and terminology are used in the papers. I did not feel the necessity of imposing a unifying standard. (After all, what is preferable: "structures" or "systems"?) Hopefully, the reader will easily overcome this lack of uniformity.

B.A. Trakhtenbrot