Lecture Notes in Computer Science

Edited by G. Goos and J. Hartmanis

454

Volker Diekert

Combinatorics on Traces

Springer-Verlag
Berlin Heidelberg New York London
Paris Tokyo Hong Kong Barcelona

Author
Volker Diekert
Institut für Informatik, Technische Universität München
Postfach 20 24 20, D-8000 München 2, FRG

CR Subject Classification (1987): F.m, F.3.2, F.4.2–3, G.2.1

ISBN 3-540-53031-2 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-53031-2 Springer-Verlag New York Berlin Heidelberg

Printed in Germany

Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
2145/3140-543210 – Printed on acid-free paper

Für
Christine, Florian, Nikolai
und Michael

Foreword

Research on formal methods for the description and analysis of the behavior of nonsequential systems has increased considerably in the last decade. This has been mainly due to the rapidly growing importance of multiprocessor configurations, distributed systems and communication networks. Another important factor of acceleration was that a number of older lines of research in informatics, mathematics and logic appeared to fit together when seen under the new aspects of distributedness, concurrency and communication, and that the new theoretical results were greatly needed by practitioners in order to cope with the complexity of nonsequential systems.

There are two main approaches to a theory of nonsequential systems. The older one, created by C.A. Petri at the beginning of the 1960s, started out from the classical theory of sequential machines but uses as basic notions concurrency and asynchronous communication between distributed activities (which causes local state transformations). The fundamental concept is nonsequentiality, and sequential systems are simply the special cases where concurrency does not occur. The formal system model of this theory, the Petri net, was soon used widely in applications and inspired many theoretical investigations. The development of the Petri-net-related theory was particularly quick and fruitful when deeper connections to classical theories like those of formal languages, semilinear sets and vector addition systems were discovered and intensively used.

The second approach, developed at the end of the 1970s by R. Milner and C.A.R. Hoare, was more programming language oriented and started out from the ideas around λ-calculus; it is based on the synchronous communication of sequential processes and was the basis for several programming languages and influential theories for the specification and semantics of communicating sequential systems.

One of the reasons for the great success of the Milner-Hoare theory is that it stays very close to the traditional concept of sequential systems. This pertains not only to the structure (non-sequential systems are obtained as parallel compositions of sequential components) but also to the semantics: the behavior of a system is described from the point of view of one single sequential observer, who can only note sequences of actions or events. Thus concurrent activities of the system are noted in an arbitrary order, i.e. concurrency is modeled by nondeterminism. This allows us to use the large body of knowledge from formal language and automata theory and from logics and semantics of sequential programming languages. But it cannot cope adequately with all phenomena of nonsequential systems.

In Petri's approach a clear distinction between nondeterminism and concurrency is made: Concurrency is considered as a nontransitive relation, in particular it is different from simultaneity. Its complement is describing causal dependencies. Therefore the semantics is technically more complicated. Observations have to include the dependency of activities, i.e. are partially ordered structures for which not much classical theory previously existed. Although considerable progress has been made in this area, the mathematical treatment of partial order semantics in its full generality is quite cumbersome, such that it is worthwile to look for compromises between the two extremes.

The most successful idea for distinguisting concurrency from nondeterminism without departing too much from the well developed and nice sequential theory came from A. Mazurkiewicz as early as 1977. His idea simply is to endow the sequential observer of a nonsequential system with a small amount of information about the structure of the system, namely its concurrency relation. This, however, is only meaningful for unlabelled systems, where actions at different positions have different names, and all are observable. It soon turned out that this nice trick associates well-known mathematical objects with nonsequential systems: the free partially commutative monoids, which are indeed intermediate between the free (noncommutative) monoids used in the Milner-Hoare theory and the free (totally) commutative monoids occuring when Petri nets are viewed from the standpoint of sequential automata theory (namely as vector addition systems).

Following Mazurkiewicz's terminology, the theory of free partially commutative monoids is called *trace theory* when it is used for the semantics of nonsequential systems; these monoids are then called trace monoids and their elements traces since they denote traces of processes in nonsequential systems. Trace theory and other investigations of free partially commutative monoids have shown a vivid development over the last few years. It became clear that trace theory is mathematically deep and beautiful, semantically powerful and flexible, and a very challenging part of theoretical informatics. This was particularly obvious at the international workshop on free partially commutative monoids organized by V. Diekert as part of the activities of the ESPRIT Basic Research Action No. 3166 "Algebraic and Syntactic Methods in Computer Science (ASMICS)" in October 1989 in Kochel am See, Bavaria, see [Die90c] and [Die90d].

V. Diekert has contributed considerably to this development in three different ways, which are brought together in this LNCS volume. He presents and extends the algebraic and combinatorial foundations of trace theory in a coherent way and embeds it in or relates it to other mathematical or informatical theories. He develops a new theory of replacement systems especially for trace monoids, and he enlarges the range of application of the theory to a much wider class of Petri nets than that considered previously.

This text is self-contained apart from basic theoretical informatics and is written in an easily readable form such that it may serve as a textbook for a first-year graduate course.

Munich, April 1990 Wilfried Brauer

Preface

These notes are a treatise on partially commutative words, commonly called traces. The underlying algebraic structures are free partially commutative monoids which have been introduced by P. Cartier and D. Foata in order to solve some combinatorial problems of rearrangement. In computer science they have been recognized as an algebraic model for concurrency. This is essentially due to the work of A. Mazurkiewicz who also introduced the basic notions and the word *trace*. Mazurkiewicz used traces as a partial order semantics for safe Petri nets. Since his initiating work a systematic study of traces under various aspects has begun.

The present volume is an extended and revised version of the *Habilitationsschrift* of the author written at the Technical University of Munich. It contains some basic material on traces including Ochmanski's characterization of recognizable trace languages and Zielonka's theory of asynchronous automata. The third chapter is devoted to an application of this theory to a modular approach for the computation of Petri net languages. This is based on so-called local morphisms between nets which yield certain homomorphisms between the corresponding trace monoids and allows a convenient treatment of the synchronization of nets. Another part of these notes concern the Möbius function and its relations to semi-Thue systems. In the last chapter we generalize the concept of semi-Thue systems to a combinatorial theory of rewriting on traces. This can be viewed as an abstract calculus for transformations of concurrent processes.

Acknowledgements
These notes would have not been written without Professor W. Brauer. I wish to thank him for constant encouragement and support over the last few years. Many fruitful discussions with him stimulated the progress of my work and led to substantial improvements.

Many thanks also to Professors M. Paul and D. Perrin who were further referees of my *Habilitationsschrift*.

I am also greatly indebted to all members of the group *Theoretische Grundlagen der Informatik* at Hamburg and Munich. In particular, I am grateful to Heino Carstensen, Jörg Desel, Matthias Jantzen, Dirk Hauschildt, Astrid Kiehn, Manfred Kunde, Klaus-Jörn Lange, Oliver Schoett, Dirk Taubner and Walter Vogler for various exchanges of ideas.

This work has also benefited very much from discussions with my colleagues: Ron Book, Christian Choffrut, Robert Cori, Christine Duboc, Zoltán Ésik, Do-

minique Foata, Massimiliano Goldwurm, Hendrik-Jan Hoogeboom, Roman König, Giancarlo Mauri, Ernst Mayr, Antoni Mazurkiewicz, Yves Métivier, Maurice Nivat, Edward Ochmanski, Friedrich Otto, Dominique Perrin, Jean-Eric Pin, Nicoletta Sabadini, Jacques Sakarovitch, Volker Strehl, Klaus Reinhardt, Grzegorz Rozenberg, Wolfgang Thomas, Celia Wrathall and Wieslaw Zielonka.

Many of them are gathered in the EBRA-working group No. 3166 "Algebraic and Syntactic Methods in Computer Science (ASMICS)" and/or in the EBRA-project No. 3148 "Design Methods Based on Nets (DEMON)". I thank these ESPRIT Basic Research Actions, ASMICS and DEMON, for all kinds of support which helped me to complete this work.

Last but not least I thank Harald Hadwiger for his excellent work in transforming the handwritten manuscript into LaTeX and I thank Springer-Verlag for accepting it into its Lecture Notes series.

Munich, April 1990 — Volker Diekert

Contents