Lecture Notes in Computer Science 922
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Advisory Board: W. Brauer D. Gries J. Stoer

Heiko Dörr

Efficient Graph Rewriting and Its Implementation

Springer

Series Editors

Gerhard Goos
Universität Karlsruhe
Vincenz-Priessnitz-Straße 3, D-76128 Karlsruhe, Germany

Juris Hartmanis
Department of Computer Science, Cornell University
4130 Upson Hall, Ithaca, NY 14853, USA

Jan van Leeuwen
Department of Computer Science, Utrecht University
Padualaan 14, 3584 CH Utrecht, The Netherlands

Author

Heiko Dörr
Institut für Informatik, Freie Universität Berlin
Takustrasse 9, D-14195 Berlin, Germany

Library of Congress Cataloging-in-Publication Data. Dörr, Heiko, 1962– Efficient graph rewriting and its implementation/Heiko Dörr. p.cm. – (Lecture notes in computer science; 922) Includes bibliographical references and index. ISBN 3-540-60055-8 (alk. paper). – ISBN 0-387-60055-8 (U.S.: alk. paper). 1. Rewriting systems (Computer science) I. Title. II. Series.
QA267. D67 1995

005. 13' 1–dc20 95-30326 CIP

CR Subject Classification (1991): F.4.2, D.3.2, D.3.4, D.2.1

ISBN 3-60055-8 Springer-Verlag Berlin Heidelberg New York

CIP data applied for

Typesetting: Camera-ready by author
SPIN: 10485935 06/3142-543210 - Printed on acid-free paper

Preface

Graph rewriting systems have come of age. In autumn 1994, the 25th anniversary of the first publication in this area was celebrated at the 5th Workshop on Graph Grammars and their Applications to Computer Science. In the interim, the subject has evolved. The current situation can be described by a three-stage model. At the very low level there is a common idea of graph rewriting as the basic mechanism, where a graph is transformed by the application of a rewriting rule.

In the second stage, this mechanism is expressed in several ways. Usually, two so-called approaches are distinguished: the algorithmic (or set-theoretic) and the algebraic approach. Both provide a formalisation of graph rewriting. They give a precise semantics to the idea of graph transformation and, hence, allow for a formal treatment. In that sense, they are similar to the semantics of programming languages.

The upper stage is partitioned into several branches. At one extreme, theoretical studies on the generational power, on semantic constructs, or on restricted formalisms are undertaken. At the other extreme, specifications of real-world systems, or implementations of rewriting environments are developed. Because the individual problems to solve are complicated enough, the branches are not very aware of each other.

The monograph builds bridges between various areas of interest: 1) it studies a class of graph rewriting systems which is very suitable for an efficient execution; 2) it presents a compilation approach to the implementation of an environment for graph rewriting; 3) it develops an implementation of a functional programming language to show that and how the presented ideas apply to real-world problems.

This publication is my dissertation which I submitted to the *Fachbereich Mathematik und Informatik* at the *Freie Universität Berlin*. It was finished in December 1994, and presents the results of the research carried out at the *Institut für Informatik*. In that respect I would like to thank several people who accompanied my work. Prof. Elfriede Fehr provided an excellent environment for my studies and gave me any support I requested; Prof. Gregor Engels introduced me to graph rewriting and encouraged my progress with valuable comments; the GraGra-AG, in particular Prof. Hartmut Ehrig and Gabi Taenzer, provided a refuge from the graph rewriting Diaspora; Prof. Raúl Rojas posed the very question initiating the whole theoretical consideration; Albert Zündorf and Andreas Schürr provided me with the newest versions of PROGRES and the additional information which was not included in the documentation; my colleagues at work provided an important social background; Elke Kasimir implemented most parts of the environment and was a very valuable critic;

Peter Hofmann developed a compiler for functional languages based on the ideas presented in Chapter 6; Gaye Rochow and John Kelly helped me by proof-reading the manuscript to eliminate the mistakes made by a non-native speaker: Thanks to you all.

Berlin, April 1995 Heiko Dörr

Contents