P. LESCANNE

CRD

CORPORATE RESEARCH AND DEVELOPMENT · SCHENECTADY, NEW YORK

PROCEEDINGS OF AN NSF WORKSHOP ON THE REWRITE RULE LABORATORY

SEPTEMBER 6-9, 1983

by
J.V. Guttag*, D. Kapur, and D.R. Musser
Information Systems Laboratory

TECHNICAL INFORMATION SERIES

CLASS 1

Report No. 84GEN008 April 1984

*Laboratory for Computer Science, MIT,
Cambridge, MA 02139

GENERAL ELECTRIC

TABLE OF CONTENTS

Section		Page
	Introduction.	1
I.	**REWRITE RULE LABORATORY**	
	REVE: A Term Rewriting System Generator with Failure-Resistant Knuth-Bendix R. Forgaard and J.V. Guttag	5
	Experiments with and Architecture of RRL, A Rewrite Rule Laboratory D. Kapur and G. Sivakumar	33
	The L Programming Language Preliminary Reference Manual D.R. Musser	57
II.	**UNIFICATION**	
	A Rehabilitation of Robinson's Unification Algorithm J. Corbin and M. Bidoit	73
	Associative-Commutative Unification F. Fages	75
	A New Equational Unification Method: A Generalisation of Martelli-Montanari's Algorithm C. Kirchner	77
	Abelian Group Unification Algorithms for Elementary Terms D. Lankford, G. Butler and B. Brady	101
III.	**TERMINATION ORDERINGS**	
	How to Prove Termination? An Approach to the Implementation of a New Recursive Decomposition Ordering P. Lescanne	109
	An Associative Path Ordering D.A. Plaisted	123
IV.	**COMPLETION PROCEDURES**	
	A Progress Report on New Decision Algorithms for Finitely Presented Abelian Groups D. Lankford, G. Butler, and A. Ballantyne	137
	A Completion Procedure for Globally Finite Term Rewriting Systems R. Gobel	155
	Completion of a Set of Rules Modulo a Set of Equations J.P. Jouannaud and H. Kirchner	207

TABLE OF CONTENTS (continued)

Section		Page
V.	**APPLICATIONS OF TERM REWRITING METHODS**	
	TeRSe: A Term Rewriting Theorem Prover J. Hsiang and N.A. Josephson	229
	Proof by Consistency D. Kapur and D.R. Musser	245
	Computing with Rewrite Systems N. Dershowitz	269
	A New Completeness Test J.J. Thiel	299
	ECOLOGISTE, A System to Make Complete and Consistent Specifications Easier L. Barros and J. L. Remy	301
	Rewrite Rule Based Program Transformation M. Srivas	319
	A Natural Proof System Based on Rewriting Techniques D. Kapur and B. Krishnamurthy	337

Theoretical Work

While accessing and understanding the relevant literature presents less of a problem to the theoretically-oriented than to those interested primarily in applications, the problems common to the production of all software are likely to present more of a problem. Certainly, the investment of considerable amounts of time in software development represents a serious digression for this group. Unfortunately, there are at least two excellent reasons why such a digression may seem useful or even necessary.

First, the manipulation of examples plays a vital role in much of the theoretical work in the rewrite rule area. Before trying to prove a difficult conjecture one often spends some time looking for a counterexample. If one finds one, it may indicate a useful way to "patch" the conjecture. At the very least it spares one the trouble of trying to prove a false conjecture. If one doesn't find one, an examination of why the examples tried were not counter-examples is often very helpful in constructing a proof of the validity of the conjecture. In a similar vein, one often develops new conjectures through the study of examples. It is sometimes possible to work these examples by hand, but doing so is generally too difficult to consider. The alternative of writing a program to experiment with an unproven idea is also usually seen as being prohibitively time-consuming.

The second reason is that it is difficult to judge the utility of much of the work in this area. Decision procedures don't exist for deciding most of the important questions about a rewriting system; e.g., is it terminating, is it confluent, is this or that theorem in its theory, etc. Consequently, a great deal of effort has been devoted to the development of restricted classes of rewriting systems for which some questions are decidable, and to the development of semi-decision procedures for unrestricted sets of rewrite rules. The utility of such work often hinges on whether it deals with a significant sub-set of those sets of rules that arise in various applications. Even when a technique is in principle applicable to a wide class of rewriting systems, efficiency issues often arise. The worst case running time of many important procedures and algorithms is clearly prohibitive. This leads one to consider average case behavior. However, meaningful analytic results in this area can be exceedingly difficult to derive. One may have to consider such things as the number of rules, the size of the rules, the structure of the rules, etc. In many cases, a procedure's primary use is as a subroutine of some other procedure, and its efficiency is most productively studied in a specialized context established by the calling procedure.

The difficulty of judging the utility of new procedures and algorithms leads one to attempt empirical evaluation. Unfortunately, it is usually impossible to conduct useful experiments by hand. One has the choice of either implementing one's techniques and trying them on an appropriate data base of examples (which one will probably have to create), or merely speculating on the applicability of those techniques. Researchers in the field, confronted with the difficulty of doing the former, have almost invariably chosen the latter.

Rewrite Rule Laboratory – REVE and RRL

To address the above-mentioned needs, a Rewrite Rule Laboratory is being jointly developed by the Computer Science Branch of General Electric Corporate Research and Development (GECRD) and MIT's Laboratory for Computer Science, supported by a grant from the National Science Foundation and a contract from the Defence Advanced Research Projects Agency. Through the Rewrite Rule Laboratory, we hope to provide an environment consisting of a portable rewrite rule virtual machine; commonly used algorithms for manipulating term rewriting systems, such as the Knuth-Bendix completion procedure and algorithms for proving termination; a language for use in small experiments with rewrite rule algorithms and a data base of examples. This environment can be used by researchers to experiment with old algorithms and quickly code new algorithms in order to try them on examples. We also hope that the environment will provide a firm base upon which to build application systems that make extensive use of term rewriting methods.

A preliminary design of the laboratory was completed and implemented in CLU at MIT by Pierre Lescanne in 1982. REVE-1 was made available to a restricted community of researchers active in the area of term rewriting systems, primarily those based at MIT, GECRD, Centre de Researche en Informatique de Nancy (CRIN), Rensselaer Polytechnic Institute (RPI), State University of New York at Stony Brook, and the University of Illinois. REVE was redesigned, primarily by Randy Forgaard, in the summer of 1983. REVE-2 provides more functionality than REVE-1 and has a better user interface. The first article in this proceedings describes in detail the design of REVE-2.

RRL is a Rewrite Rule Laboratory written in Lisp and L under development at GECRD and RPI. The work on RRL was started following the successful prototype implementation of REVE-1 and considerable experimentation with it. The second article in the proceedings discusses the architecture of RRL and experiments done on RRL on a number of examples.

A programming language, L, based on LOGO and LISP, is being developed at GECRD to serve as the language with which a user programs small experiments. It is intended to be used as the command language component of the user interface of the Rewrite Rule Laboratory both for REVE and RRL. The third article in the proceedings is an introduction to the L language. The fourth article summarizes an experiment with a unification algorithm done in L.

We hope that REVE, RRL, and subsequent versions of the rewrite rule laboratory that evolve from them, will facilitate the conducting of experiments with rewriting systems, and will provide a firm base upon which to build application programs.

Workshop on Rewrite Rule Laboratory

A workshop on the rewrite rule laboratory was held at General Electric Corporate Research and Development Center (GECRD), Schenectady, New York, and in Rensselaerville, New York, from September 4-6, 1983. A group of researchers, most of whom had been using REVE-1, were invited to the worship to provide their feedback on the designs of REVE, RRL, and L, as well as to coordinate future work. The participants were:

D. Plaisted, University of Illinois - Urbana-Champaign,

D. Lankford, Louisiana Tech University, Ruston, Louisiana,

J. Hsiang and M.K. Srivas, State University of New York - Stony Brook,

D. Benanav, M.S. Krishnamoorthy and R.F. McNaughton, P. Narendran, and G. Sivakumar of Rensselaer Polytechnic Institute, Troy, NY,

R. Gobel, University Kaiserslautern, West Germany,

J. Corbin of Laboratoires de Marcoussis, Centre de Recherche de la C.G.E., France.

J.-P. Jouannaud, H. Kirchner, C. Kirchner, P. Lescanne, and J.-L. Remy of Centre de Researche en Informatique de Nancy, France,

R. Forgaard, J. Guttag, and K. Yelick of MIT, and

R. Harris, D. Kapur, B. Krishnamurthy, and D. Musser of GECRD.

On the first day of the workshop, demonstrations of the REVE system were given by Deepak Kapur and Randy Forgaard. There were presentations on other existing systems based on the rewrite rule approach. Claude Kirchener discussed his experience with the FORMEL system of INRIA. David Plaisted summarized the OBJ system developed by J. Goguen's group at SRI. Richard Gobel presented an overview of the SEKI project and Rewrite Rule Laboratory under development in the SEKI project. David Musser gave an overview of the AFFIRM system.

On the second day, Helene Kirchner presented a design of the generalized Knuth-Bendix completion procedure which the CRIN group was planning to implement in REVE. Randy Forgaard discussed the new design of REVE and the user interface. Deepak Kapur presented the design of RRL. David Musser gave an overview of the L language.

The workshop also provided an excellent opportunity for the participants to discuss and exchange recent theoretical results on term rewriting systems. The third day and the morning of the fourth day was devoted to technical presentations. The remaining part of the fourth day was used to coordinate future work on the implementations of the REVE and RRL systems.

The first four articles discuss REVE, RRL, and L. Except for the paper by Dershowitz, the rest of the proceedings is a collection of papers presented at the workshop. The paper by Thiel was presented by Pierre Lescanne and the paper by Fages was presented by Jean-Pierre Jouannaud. Dershowitz's paper was not actually presented at the workshop, since he could not attend.

The papers by Corbin and Bidoit, Fages, and Thiel have appeared elsewhere, so only abstracts are included. We expect that other papers in the proceedings will eventually appear in conference proceedings and journals. We hope that the publication of this proceedings will aid in early dissemination of useful results.

John Guttag
Deepak Kapur
David Musser

March 30, 1984