Andrzej M. Borzyszkowski
Stefan Sokołowski (Eds.)

Mathematical Foundations of Computer Science 1993

18th International Symposium, MFCS'93
Gdańsk, Poland, August 30-September 3, 1993
Proceedings

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

Series Editors

Gerhard Goos
Universität Karlsruhe
Postfach 69 80
Vincenz-Priessnitz-Straße 1
D-76131 Karlsruhe, Germany

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

Volume Editors

Andrzej M. Borzyszkowski
Stefan Sokołowski
Institute of Computer Science, Polish Academy of Sciences
ul. Jaskowa Dolina 31, 80-252 Gdańsk, Poland

CR Subject Classification (1991): D.3.3, F.1.1, F.2.2, F.3.1, F.4.1, F.4.3, D.2.1-2, G.2.2

ISBN 3-540-57182-5 Springer-Verlag Berlin Heidelberg New York
ISBN 0-387-57182-5 Springer-Verlag New York Berlin Heidelberg

Typesetting: Camera-ready by authors
Printing and binding: Druckhaus Beltz, Hemsbach/Bergstr.
45/3140-543210 - Printed on acid-free paper

Preface

This volume contains the proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science, MFCS'93, held in Gdańsk, Poland, August 30–September 3, 1993.

The MFCS symposia, organized annually in Poland and the former Czechoslovakia since 1972, have a long and well-established tradition. Over the years they have served as a meeting ground for specialists from all branches of theoretical computer science, in particular

algorithms and complexity, automata theory and theory of languages,
concurrent, distributed and real-time systems,
the theory of functional, logic and object-oriented programming,
lambda calculus and type theory,
semantics and logics of programs,

and others.

The interest in MFCS symposia seems to be growing. MFCS'93 sets a new record on the number of submissions: 133. Out of them, the Programme Committee has selected 56 for presentation, strictly on the basis of their scientific merit and appropriateness for the symposium. The committee was assisted by over 210 referees. The scientific programme includes also lectures by 12 distinguished scholars.

The proceedings contain both the invited lectures and the contributions. Unfortunately, most of the latter had to be abridged due to space limitations.

We would like to thank, first of all, the authors of the papers submitted—their interest justifies the organization of the symposia. Thanks are also due to the members of the Programme Committee and to the referees for their work put into the evaluation of the papers. Finally, we thank Springer-Verlag for a long-term cooperation with MFCS symposia.

A number of institutions have financially contributed to the success of the symposium. We gratefully acknowledge the support from Stefan Batory Foundation, *Gdańsk Development Method* research project, Warsaw University, Wroclaw University, Gdańsk University.

MFCS'93 is organized by the Institute of Computer Science of the Polish Academy of Sciences. The Organizing Committee consists of Marek A. Bednarczyk (chairman), Andrzej M. Borzyszkowski, Ryszard Kubiak, Wiesław Pawłowski and Stefan Sokołowski.

Gdańsk, June 1993

Andrzej M. Borzyszkowski
Stefan Sokołowski

Programme Committee

J. Adámek (Praha)
M. A. Bednarczyk (Gdańsk)
J. A. Bergstra (Amsterdam)
A. M. Borzyszkowski (Gdańsk)
B. Courcelle (Bordeaux)
H.-D. Ehrich (Braunschweig)
I. Havel (Praha)
G. Huet (Paris)
N. Jones (København)
E. Moggi (Genova)
V. Nepomniaschy (Novosibirsk)
E.-R. Olderog (Oldenburg)
F. Orejas (Barcelona)
A. Poigné (St. Augustin)
I. Sain (Budapest)
D. A. Schmidt (Manhattan, Kansas)
S. Sokołowski (Gdańsk), chair
M. Wirsing (München)

Non-PC Referees

K. Ambos-Spies
N. Andersen
H. R. Andersen
H. Andréka
M. Anselmo
P. Audebaud
J. L. Balcazar
A. Banerjee
P. Barth
M. Bauderon
T. Bétréma
U. Berger
J. Berstel
J. Biskup
A. J. Blikle
J. Bradfield
F. J. Brandenburg
R. Breu
C. Brown
F. Broy
A. Bystrov
R. Casas
I. Castellani
M. V. Cengarle
M. M. Chiara
C. Choffrut
J. Chomicki
J. Clausen
L. Colson
T. Coquand
G. Costa
L. Csirmaz
P.-L. Curien
L. Czaja
R. De Nicola
P. Dembiński
M. Dezani
R. Diaconescu
J. Diaz
V. Diekert
G. Dowek
D. Dranidis
P. Dybjer
H. Ehrig
V. Evstigneev
G.-L. Ferrari
J. L. Fiadeiro
H. Fleischhack
W. J. Fokkink
G. S. Frandsen
D. de Frutos-Escrig
J. Gabarro
A. Gammelgaard
G. Ghelli
P. Y. Gloess
L. Godo
M. Gogolla
J. Goguen
M. Grosse-Rhode
R. Grosu
C. A. Gunter
P. Hajnal
M. R. Hansen
T. Hardin
J. Hatcliff
M. Hennessy
M. Hermann
R. Herzig
L. Holenderski
R. Howell
J. Hromkovič
M. Huhn
H. Hungar
G. Jarzembski
J. Jędrzejowicz
B. Jenner
C. T. Jensen
J. Jørgensen
B. Josko
J.-P. Jouannaud
A. Jung
S. Kaes
G. Kahn
R. Kaivola
J. Kari
S. Kars
J. Katajainen
C. Kenyon
D. Kesner
N. Klarlund
J. W. Klop
S. Klusener
B. Konikowska
S. Kotov
V. Koubek
H.-J. Kreowski
B. Krieg-Brückner
M. Křivánek
F. Kröger
R. Kubiak
A. Kučera
G. Kucherov
Á. Kurucz
H. Le Verge
U. Lechner
C. Lengauer
P. Lincoln
U. W. Lipeck
I. Litovsky
G. Longo
A. Lozano
Z. Luo
T. Maibaum
K. Malmkjær
V. Manca
S. Martini

E. W. Mayr
N. D. N. Measor
S. Meiser
V. Ménissier-Morain
A. Middeldorp
S. Mikulás
P. B. Miltersen
T. Mogensen
B. Monien
L. Moss
P. D. Mosses
A. W. Mostowski
F. Mráz
A. Mück
D. Murphy
I. Németi
M. Nielsen
R. Nieuwenhuis
U. Nilsson
E. Ohlebusch
L. Ong
E. Orłowska
Y. Ortega-Mallén
L. Pacholski
P. Pączkowski
C. Palamidessi
C. Paulin
W. Pawlowski
W. Penczek
H. Peterreins
W. Phoa
B. Pierce
G. M. Pinna
D. Pisinger
A. Pitts
V. Pollara
J. Power
T. Przytycka
Z. Qian
A. Raspaud
G. Rauzy
G. Reggio
J. Rehof
W. Reisig
B. Reus
J. Roman
M. Rosendahl
G. Rosolini
B. Rovan
M. Rodriguez-Artalejo
M. Rusinowitch
G. Saake
N. Saheb
A. Salomaa
D. Sands
M. Schenke
G. Schreiber
M. Schwartzbach
A. Schwill
G. Scollo
G. Serény
M. Serna
H. Shi
N. V. Shilov
A. Simon
G. Snelting
S. Soloviev
E. Sopena
Z. Spławski
M. Srebrny
R. Statman
O. Stepankova
T. Streicher
W. Struckmann
A. Sulimov
A. Szałas
A. Szepietowski
A. Tarlecki
E. Teniente
J. Toran
J. L. Träff
E. Upfal
P. Urzyczyn
Y. Venema
R. Verbeek
A. Vinacua
I. B. Virbitskaite
S. Vorobyov
D. Wätjen
K. Weihrauch
B. Werner
J. Winkowski
P. Winter
M. Zaionc
E. Zucca
A. Zvonkin
J. Zwiers

A couple of referees have remained anonymous to the editors of the volume. We are as thankful to them as to the ones listed above for their work.

Table of Contents

Invited lectures

Contributions

On the Unification Free Prolog Programs

Krzysztof R. Apt[1] and Sandro Etalle[2]

[1] CWI
P.O. Box 4079, 1009 AB Amsterdam, The Netherlands
and
Faculty of Mathematics and Computer Science
University of Amsterdam, Plantage Muidergracht 24
1018 TV Amsterdam, The Netherlands
[2] Dipartimento di Matematica Pura ed Applicata
Università di Padova
Via Belzoni 7, 35131 Padova, Italy

Abstract. We provide simple conditions which allow us to conclude that in case of several well-known Prolog programs the unification algorithm can be replaced by iterated matching. The main tools used here are types and generic expressions for types. As already noticed by other researchers, such a replacement offers a possibility of improving the efficiency of program's execution.

Notes. The work of the first author was partly supported by ESPRIT Basic Research Action 6810 (Compulog 2). This research was done partly during the second author's stay at Centre for Mathematics and Computer Science, Amsterdam.

1 Introduction

Unification is heralded as one of the crucial features offered by Prolog, so it is natural to ask whether it is actually used in specific programs. The aim of this paper to identify natural conditions under which unification can be replaced by iterated matching and to show that they are applicable to several well-known Prolog programs. These conditions can be statically checked without analyzing the search trees for the queries. For programs which use ground inputs they can be efficiently tested.

The problem of replacing unification by iterated matching was already studied in the literature by a number of researchers – see e.g. Deransart and Maluszynski [DM85b], Maluszynski and Komorowski [MK85] and Attali and Franchi-Zannettacci [AFZ88]. As in the previous works on this subject, we use modes, which indicate how the arguments of a relation should be used. Our results improve upon the previous ones due to the additional use of types. This allows us to deal with non-ground inputs.

We use here a simple notion of a type, which is a set of terms closed under substitution. The main tool in our approach is the concept of a *generic expression*. Intuitively, a term s is a generic expression for a type T if it is more general than all elements of T which unify with s. This simple notion turns out to be crucial here, because surprisingly often the input positions of the heads of program clauses are filled in by generic expressions for appropriate types.

We combine in our analysis the use of generic expressions with the notion of a *well-typed program*, recently introduced by Bronsard, Lakshman and Reddy [BLR92], which allows us to ensure that the input positions of the selected atoms remain correctly typed. As the table included at the end of this paper shows, our results can be applied to astonishingly many Prolog programs.

2 Preliminaries

In what follows we study logic programs executed by means of the *LD-resolution*, which consists of the SLD-resolution combined with the leftmost selection rule. An SLD-derivation in which the leftmost selection rule is used is called an *LD-derivation*. We allow in programs various first-order built-in's, like $=$, $\neq$, $>$, etc, and assume that they are resolved in the way conforming to their interpretation.

We work here with *queries*, that is sequences of atoms, instead of *goals*, that is constructs of the form $\leftarrow Q$, where Q is a query. Apart from this we use the standard notation of Lloyd [Llo87] and Apt [Apt90]. In particular, given a syntactic construct E (so for example, a term, an atom or a set of equations) we denote by $Var(E)$ the set of the variables appearing in E. Given a substitution $\theta = \{x_1/t_1, ..., x_n/t_n\}$ we denote by $Dom(\theta)$ the set of variables $\{x_1, \ldots, x_n\}$, by $Range(\theta)$ the set of terms $\{t_1, \ldots, t_n\}$, and by $Ran(\theta)$ the set of variables appearing in $\{t_1, \ldots, t_n\}$. Finally, we define $Var(\theta) = Dom(\theta) \cup Ran(\theta)$.

Recall that a substitution θ is called *grounding* if $Ran(\theta)$ is empty, and is called a *renaming* if it is a permutation of the variables in $Dom(\theta)$. Given a substitution θ and a set of variables V, we denote by $\theta|V$ the substitution obtained from θ by restricting its domain to V.

2.1 Unifiers

Given two sequences of terms $\mathbf{s} = s_1, ..., s_n$ and $\mathbf{t} = t_1, ..., t_n$ of the same length we abbreviate the set of equations $\{s_1 = t_1, ..., s_n = t_n\}$ to $\{\mathbf{s} = \mathbf{t}\}$ and the sequence $s_1\theta, ..., s_n\theta$ to $\mathbf{s}\theta$. Two atoms can unify only if they have the same relation symbol. With two atoms $p(\mathbf{s})$ and $p(\mathbf{t})$ to be unified we associate the set of equations $\{\mathbf{s} = \mathbf{t}\}$. In the applications we often refer to this set as $p(\mathbf{s}) = p(\mathbf{t})$. A substitution θ such that $\mathbf{s}\theta = \mathbf{t}\theta$ is called a *unifier* of the set of equations $\{\mathbf{s} = \mathbf{t}\}$. Thus the set of equations $\{\mathbf{s} = \mathbf{t}\}$ has the same unifiers as the atoms $p(\mathbf{s})$ and $p(\mathbf{t})$.

A unifier θ of a set of equations E is called a *most general unifier* (in short *mgu*) of E if it is more general than all unifiers of E. An mgu θ of a set of equations E is called *relevant* if $Var(\theta) \subseteq Var(E)$.

The following lemma was proved in Lassez, Marriot and Maher [LMM88].

Lemma 1. *Let θ_1 and θ_2 be mgu's of a set of equations. Then for some renaming η we have $\theta_2 = \theta_1\eta$.* □

Finally, the following well-known lemma allows us to search for mgu's in an iterative fashion.