

Handbook of Logic in Artificial Intelligence and Logic Programming

Volume 2
Deduction Methodologies

Edited by
DOV M. GABBAY, C. J. HOGGER,
and J. A. ROBINSON

OXFORD SCIENCE PUBLICATIONS

TP18-62

H 236.2

V.2

9462992

HANDBOOK OF LOGIC IN ARTIFICIAL INTELLIGENCE AND LOGIC PROGRAMMING

Volume 2

Deduction Methodologies



E9462992

Edited by

DOV M. GABBAY

and

C. J. HOGGER

*Imperial College of Science, Technology and Medicine
London*

and

J. A. ROBINSON

Syracuse University, New York

Volume Co-ordinator

J. SIEKMANN

DFKI/University of Saarland, Saarbrücken



CLARENDON PRESS • OXFORD

1994

Oxford University Press, Walton Street, Oxford OX2 6DP

*Oxford New York Toronto
Delhi Bombay Calcutta Madras Karachi
Kuala Lumpur Singapore Hong Kong Tokyo
Nairobi Dar es Salaam Cape Town
Melbourne Auckland Madrid
and associated companies in
Berlin Ibadan*

Oxford is a trade mark of Oxford University Press

*Published in the United States by
Oxford University Press Inc., New York*

© The contributors listed on pp xv, 1994

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, without the prior permission in writing of Oxford University Press. Within the UK, exceptions are allowed in respect of any fair dealing for the purpose of research or private study, or criticism or review, as permitted under the Copyright, Designs and Patents Act, 1988, or in the case of reprographic reproduction in accordance with the terms of licences issued by the Copyright Licensing Agency. Enquiries concerning reproduction outside these terms and in other countries should be sent to the Rights Department, Oxford University Press, at the address above.

This book is sold subject to the condition that it shall not, by way of trade or otherwise, be lent, re-sold, hired out or otherwise circulated without the publisher's prior consent in any form of binding or cover other than that in which it is published and without a similar condition including this condition being imposed on the subsequent purchaser.

A catalogue record for this book is available from the British Library

*Library of Congress Cataloging in Publication Data
(Data available)*

ISBN 0 19 853746 8

Typeset using LaTeX by Jane Spurr

*Printed in Great Britain by
Bookcraft (Bath) Ltd
Midsomer Norton, Avon*

HANDBOOK OF LOGIC
IN ARTIFICIAL INTELLIGENCE
AND LOGIC PROGRAMMING

Editors

Dov M. Gabbay, C. J. Hogger, and J. A. Robinson

HANDBOOKS OF LOGIC IN COMPUTER SCIENCE
and
ARTIFICIAL INTELLIGENCE AND LOGIC
PROGRAMMING

Executive Editor

Dov M. Gabbay

Administrator

Jane Spurr

Handbook of Logic in Computer Science

- Volume 1** Background: Mathematical structures
- Volume 2** Background: Computational structures
- Volume 3** Semantic structures
- Volume 4** Semantic modelling
- Volume 5** Theoretical methods in specification and verification
- Volume 6** Logical methods in computer science

Handbook of Logic in Artificial Intelligence and
Logic Programming

- Volume 1** Logical foundations
- Volume 2** Deduction methodologies
- Volume 3** Nonmonotonic reasoning and uncertain reasoning
- Volume 4** Epistemic and temporal reasoning
- Volume 5** Logic programming

Preface

The preface to Volume 1 concluded with the belief that as a result of its interaction with computing and AI, logic is going to move to the next stage of its evolution. Of particular importance in accelerating this process is its attempts to cater for the needs of computationally handling human reasoning and activity.

We would like to share with the reader our vision of this evolution and interaction and how we see the handbooks as contributing to its realization.

Given an application area which may be a possible consumer for logical analysis, there are two aspects to a successful use of logic to serve the application.

The application area must be analysed and broken into distinct components, which are the moving *atomic* components of the application, ready for a suitable representation and logical analysis. The choice of atomic units may depend on the purpose and environment of the application.

Suitable representations and suitable logics need to be chosen devised and developed, which can naturally reflect the operations and movements intrinsic to the applications. In any real application there are many logics involved, formalizing different aspects of it. There are temporal aspects, modal aspects, various nonmonotonic aspects, etc. These aspects and their logics are naturally interleaved and combined in the application.

Current research activity pays a great deal of attention to methodologies of different representations and logics. An enormous amount of energy is spent on comparing logics in strength and properties and on introducing new logics, their proof theory and automated deduction. It is now time to pay more attention to how different logics and deduction methodologies can be combined. In any real AI application, the dominant mechanisms—the mechanisms reflecting *intelligence*—are the ones which combine and switch from one logic to another.

In my opinion, the very question of whether logic can successfully and fruitfully be applied in AI hinges on our ability to deliver well integrated combined multilogic systems.

To focus our thoughts, let us take a concrete example. Consider a university Head of Department planning the distribution of lectures and courses. Among other considerations, there is reasoning involving whether to hire a new temporary lecturer. Of course funds are needed but how

much funds are available depends on the success of research project applications. In practice a good candidate might be lost if not approached immediately, and so a sensible Head of Department might make an offer based on expected funds, taking the risk that no funds arrive. In fact, the university may have some guidelines on how such 'risks' may be taken.

This simple and realistic example involves several types of reasoning and automated deduction machines:

1. temporal logic;
2. resource considerations and planning;
3. modal logic;
4. two-dimensional temporal logic and updates, and its relation to actions;
5. metalevel considerations.

The modal logic has to do with considerations of possible availability of funds. The modal logic must be combined with the temporal logic with possibly some metalevel aspects, to allow us to express the guidelines. The two-dimensional temporal logic is needed to describe a possible planned future, as seen from the past, on the basis of which a past action was taken.

It is clear that the dominant logical feature for such an application is the way the pure components are combined. Even if they are all expressed in classical logic, they might not necessarily be expressed in the same deductive framework for classical logic. Resolution may be most conveniently used for the modal considerations while Horn clause logic may be used for the temporal reasoning, because it involves a strong persistence property, and hence can be formalized in the Horn fragment of classical logic.

The above shows that in fact we have to understand integration, or combination in the widest sense, not only combining different logics but also combining different deduction methodologies, of possibly the same logic.

In addition, we should regard metalevel vs object level communication as a special kind of combining logics.

I believe that the availability of good deduction methodologies for various logics will be a decisive factor in the applicability of logic in AI. These good deduction methodologies are valuable not only because of the need for practical implementable systems, but also for their role in the conceptual level, facilitating the combination of logics and logical system design.

I am therefore happy to present to you Volume 2 of the Handbook on *Deduction Methodologies*.

This volume contains chapters on the central topics of automated deduction.

Further chapters on automated deduction for non-classical logics can be found in later volumes of this Handbook, for example the chapter of

L. Fariñas del Cerro and A. Herzig on Resolution methods for temporal logics, in Volume 4.

I hope that this volume will make it easier to proceed with what I consider to be the next major step in automated deduction, that of *combining* deduction methodologies.

The Handbooks

The *Handbook of Logic in Artificial Intelligence and Logic Programming* and its companion, the *Handbook of Logic in Computer Science*, have been created in response to a growing need for an in-depth survey of the application of logic in AI and computer science.

We see the creation of the Handbook as a combination of authoritative exposition, comprehensive survey, and fundamental research exploring the underlying unifying themes in the various areas. The intended audience is graduate students and researchers in the areas of computing and logic, as well as other people interested in the subject. We assume as background some mathematical sophistication. Much of the material will also be of interest to logicians and mathematicians.

The tables of contents of the volumes were finalized after extensive discussions between Handbook authors and second readers. The first two volumes present the background logic and mathematics extensively used in artificial intelligence and logic programming. The point of view is application oriented. The other volumes present major areas in which the methods are used. These include: Volume 1—Logical foundations; Volume 2—Deduction methodologies; Volume 3—Nonmonotonic reasoning and Uncertain reasoning; Volume 4— Epistemic and temporal reasoning; Volume 5—Logic programming.

The chapters, which in many cases are of monographic length and scope, are written with emphasis on possible unifying themes. The chapters have an overview, introduction, and main body. A final part is dedicated to more specialized topics.

Chapters are written by internationally renowned researchers in their respective areas. The chapters are co-ordinated and their contents were discussed in joint meetings. Each chapter has been written using the following procedures:

1. A very detailed table of contents was discussed and co-ordinated at several meetings between authors and editors of related chapters. The discussion was in the form of a series of lectures by the authors. Once an agreement was reached on the detailed table of contents, the authors wrote a draft and sent it to the editors and to other related authors. For each chapter there is a second reader (the first reader is the author) whose job it has been to scrutinize the chapter together

with the editors. The second reader's role is very important and has required effort and serious involvement with the authors.

Second readers for this volume are:

Chapter 1: Automated reasoning—M. Stickel

Chapter 2: General unification theory—J. P. Jouannaud

Chapter 3: Induction—D. Kapur

Chapter 4: Higher-order features, types and fixpoints—A. Mycroft

Chapter 5: Metalanguages, reflection principles and self-reference—
K. Konolige and R. Weyrauch

Chapter 6: Classical vs non-classical logic—R. A. Kowalski and D. McDermott

2. Once this process was completed (i.e. drafts seen and read by a large enough group of authors), there were other meetings on several chapters in which authors lectured on their chapters and faced the criticism of the editors and audience. The final drafts were prepared after these meetings.
3. We attached great importance to group effort and co-ordination in the writing of chapters. The first two parts of each chapter, namely the introduction-overview and main body are not completely under the discretion of the author, as he/she had to face the general criticism of all the other authors. Only the third part of the chapter is entirely for the authors' own personal contribution.

The Handbook meetings were generously financed by OUP, by SERC contract SO/809/86, by the Department of Computing at Imperial College, and by several anonymous private donations.

We would like to thank our colleagues, authors, second readers, and students for their effort and professionalism in producing the manuscripts for the Handbook. We would particularly like to thank the staff of OUP for their continued and enthusiastic support, and Mrs Jane Spurr, our OUP Administrator for her dedication and efficiency.

London

November 1993

D. M. Gabbay

Contents

List of contributors	xv
----------------------	----

Logical basis for the automation of reasoning:	
Case studies	1

Larry Wos and Robert Veroff

1	Introduction	1
2	The clause language paradigm	4
	2.1 Components of the clause language paradigm	4
	2.2 Interplay of the components	6
3	A fragment of the history of automated reasoning from 1960 to 1990	8
4	Answering open questions	11
	4.1 Equivalential calculus	12
	4.2 Combinatory logic	14
	4.3 Sentential calculus	20
5	Relation to logic programming and to person-oriented reasoning	21
	5.1 Logic programming	21
	5.2 Person-oriented reasoning	24
6	Challenge problems for testing, comparing, and evaluating	25
7	Current state of the art: Basic research problems to solve	30
8	Programs, books, and a problem database	32
9	Summary and the future	35

Unification theory	41
--------------------	----

Franz Baader and Jörg Siekmann

1	Outline	41
2	What is <i>E</i> -unification?	42
3	Unification in the empty theory	47
	3.1 An informal description of Robinson's algorithm	48
	3.2 Motivation for using \emptyset -unification	49
	3.3 Efficient algorithms for \emptyset -unification	51
4	Unification in non-empty theories	56
	4.1 Motivation for using <i>E</i> -unification	58
	4.2 Single equations versus systems of equations	59
	4.3 A closer look at the signature	61
	4.4 Restricted versus unrestricted instantiation ordering	62

4.5	Notions and notation revisited	63
5	Unification results for special theories	65
6	General results	71
6.1	General E -unification	72
6.2	Semantic approaches to unification	80
6.3	The combination problem	85
7	Related areas of research	91
7.1	Higher-order unification	91
7.2	Unification in sort theories	96
7.3	Constraint solving	99
8	Applications of unification	101

Mathematical induction 127

Christoph Walther

1	Introduction	128
2	Admissible specifications and theories	132
2.1	Data structures and algorithms	132
2.2	Theorem proving	136
2.3	Induction	138
2.4	Expanded theories	141
2.5	Incomplete specifications	142
2.6	Summary	145
3	Computing induction axioms	147
3.1	Defining induction axioms	147
3.2	The induction heuristic	154
3.3	Modification of induction axioms by domain generalization	155
3.4	Comparing induction axioms	161
3.5	Modification of induction axioms by range generalization	168
3.6	Modification of induction axioms by separation	172
3.7	Summary	175
4	Proving well-foundedness	177
4.1	Termination of algorithms	177
4.2	Proving termination by induction lemmata	180
4.3	Proving termination with argument bounded algorithms	185
4.4	Computing well-founded domain and range generalizations	192
4.5	Summary	196
5	Proving induction formulas	197
5.1	The basic theorem prover	197
5.2	Symbolic evaluation	200

5.3	Applying induction hypotheses	202
5.4	Using lemmata	206
5.5	Summary	207
6	Generalizing formulas	208
6.1	Computing generalizations by inverse inference rules	208
6.2	Inverse substitution	209
6.3	Inverse functionality	213
6.4	Inverse weakening	214
6.5	Inverse splitting	216
6.6	Inverse replacement	219
6.7	Inverse modus ponens	223
6.8	Summary	225

Higher order logic 229

Daniel Leivant

1	Introduction	230
2	The expressive power of second order Logic	231
2.1	The language of second order logic	231
2.2	Expressing size	232
2.3	Defining data types	234
2.4	Describing processes	236
2.5	Expressing convergence using second order validity	237
2.6	Truth definitions: the analytical hierarchy	238
2.7	Inductive definitions	241
3	Canonical semantics of higher order logic	243
3.1	Tarskian semantics of second order logic	243
3.2	Function and relation formulations	243
3.3	Normal forms	244
3.4	Finite order logic	245
3.5	Functional types	246
3.6	Formulas as higher order functions	247
3.7	Truth definitions revisited	248
4	Proof theory	250
4.1	Basic formalisms	250
4.2	Additional set existence principles	252
4.3	Constructive finite order logics	254
4.4	Normalization and the subformula property	255
5	Ontology	257
5.1	The gulf between first order and second order logic	257
5.2	Lindström's and Quine's tests	259
5.3	Slipping from first to second order logic	261

5.4	Higher order logic as mathematics: Henkin's semantics	262
5.5	Henkin completeness for full finite order logic	265
5.6	Finite order logic as a second order theory	267
6	Restricted higher order logic	268
6.1	Restricted expressiveness 1: Monadic second order logic	269
6.2	Restricted expressiveness 2: Fixpoint logics	270
6.3	Restricted semantics: Weak second order logic	272
6.4	Predicative logic: Restricted comprehension	274
7	Mathematical practice	279
7.1	Second order axioms vs. first order schemas	279
7.2	Higher order aspects of set theory: from higher order to first order and back	282
7.3	Analysis and reductive proof theory	285
7.4	Speed-up	288
8	Higher order logic in relation to computing and programming	293
8.1	Higher order data and types	293
8.2	The computational nature of higher order natural deduction	296
8.3	Higher order logic in the meta-theory of formal systems	298
8.4	Higher order logic and computational complexity	299

Meta-languages, reflection principles, and self-reference 323

Donald Perlis and V. S. Subrahmanian

1	Introduction	323
1.1	Aims	323
1.2	Aboutness and self-reference	324
1.3	Truth and paradoxes	327
1.4	Ideal, pseudo-ideal and situated reasoning	329
2	Formal definitions and difficulties	331
2.1	Meta-languages and Tarski hierarchies	331
2.2	Reflection principles	333
2.3	Fixed points	338
3	Applications	339
3.1	Ideal and pseudo-ideal approaches (to knowledge)	339
3.2	Situated temporal reasoning and self-reference	341
3.3	Logic programming	344
3.4	Naive vs. amalgamated LP	351
4	Future developments	353

4.1	Agent referential semantics	353
5	Conclusions	355

Classical vs non-classical logics (the universality of classical logic) 359

D. M. Gabbay

1	Introduction—the debate	359
2	What is a logical system?—the challenge	366
2.1	Logical systems as consequence relations	368
2.2	Logical systems as algorithmic proof systems	371
2.3	Logical systems as algorithmic structured consequence relations	372
2.4	Logical systems as labelled deductive systems	374
2.5	Aggregated systems	382
2.6	Practical reasoning systems	384
3	How to construct a logic for an application: the case studies	385
3.1	Case study 1: temporal logic in two-sorted classical logic	388
3.2	Case study 2: priority logic and PROLOG	395
4	Algebraic LDS: a unifying solution	403
5	Reductions to classical logic: the options	427
6	Translations into classical logic: technical case study	436
7	Linked predicate languages	443
7	Linked predicate languages: classical logic as a target for translation	443
8	The meta-language HFP: computational classical logic	451
9	Semi-algebraic semantics for propositional logics	465
10	An automated universal translator into classical logic	478
10.1	The translation steps	479
10.2	The SCAN algorithm	483
11	Conclusion: the current state of the debate	495

Index 501

Logical Basis for the Automation of Reasoning: Case Studies *

Larry Wos and Robert Veroff

Contents

1	Introduction	1
2	The clause language paradigm	4
	2.1 Components of the clause language paradigm	4
	2.2 Interplay of the components	6
3	A fragment of the history of automated reasoning from 1960 to 1990	8
4	Answering open questions	11
	4.1 Equivalential calculus	12
	4.2 Combinatory logic	14
	4.3 Sentential calculus	20
5	Relation to logic programming and to person-oriented reasoning	21
	5.1 Logic programming	21
	5.2 Person-oriented reasoning	24
6	Challenge problems for testing, comparing, and evaluating	26
7	Current state of the art: Basic research problems to solve	30
8	Programs, books, and a problem database	32
9	Summary and the future	35

1 Introduction

With the availability of computers in the late 1950s, researchers began to entertain the possibility of automating reasoning. Immediately, two distinctly different approaches were considered as the possible basis for that automation. In one approach, the intention was to study and then emulate people; in the other, the plan was to rely on logic. In this chapter, we focus mainly on the latter, for logic has proved to provide an excellent basis for the automation of reasoning.

*This work was supported by the Office of Scientific Computing, U.S. Department of Energy, under Contract W-31-109-Eng-38.

Among the various paradigms for automated reasoning that rely on logic, in this chapter we concentrate on the *clause language paradigm* [Wos, 1987; Wos *et al.*, 1992]. In addition to relying on the use of clauses to represent information, the clause language paradigm—in contrast to paradigms based on logic programming—relies on the retention of deduced conclusions and on the use of a variety of inference rules, diverse strategies, and a number of additional procedures. The clause language paradigm is discussed in detail in Section 2.

The current state of the clause language paradigm, which continues to evolve, results from the objective of using automated reasoning for research in mathematics and logic. Indeed, the pursuit of that objective prompted numerous experiments that in turn revealed the need for the various components of the paradigm. To provide a perspective for measuring the progress that has occurred in the preceding decade, and to illustrate graphically the value of experimentation, we present in Section 3 experiments that directly influenced the evolution of the clause language paradigm.

The applications of automated reasoning include research in mathematics and logic, program verification and synthesis, circuit design and validation, and, in general, tasks that depend on logical reasoning. Substantial evidence exists that automated reasoning now occupies a significant position in science, for its use has led to answering open questions in finite semigroups [Winker *et al.*, 1981], in equivalential calculus [Peterson, 1977; Kalman, 1978; Wos *et al.*, 1984], in combinatory logic [Smullyan, 1985; Smullyan, 1987; Wos and McCune, 1988], and in sentential calculus [Scott, 1990]. To assess the scope and significance of the various contributions, we discuss in Section 4 a number of the questions that have been answered by relying heavily on an automated reasoning program.

To gain an insight into the possible reasons for the cited successes and also gain an appreciation for how a program based on the clause language paradigm complements the mechanisms on which people rely, we compare in Section 5 the clause language paradigm with both logic programming and person-oriented reasoning.

The long-term objectives for automated reasoning all focus on providing an automated reasoning assistant. At the more mundane end of the spectrum, the envisioned assistant can be instructed to function as a logical calculator, carrying out assignments that only barely require reasoning. At the more intriguing end of the spectrum, this assistant can be instructed to function as a colleague, self-analytically choosing inference rules and strategies, modifying an attack based on current performance, and examining results to highlight those that offer the most significance. The assistant will offer interactive mode, batch mode, and ‘collaborative mode’—the latter referring to the style that is present when working with a research colleague in which interaction occurs, but only at the highest level.

Despite the marked advances of the preceding decade, there remain many obstacles to overcome before automated reasoning reaches its full potential. In the obvious sense, each obstacle is a source of significant research in the field. Although some of the following obstacles do not directly apply to every paradigm for the automation of reasoning, each in some manner merits study.

1. **Clause Generation:** The reasoning program draws far too many conclusions, many of which are redundant and many of which are irrelevant even though they are not redundant.
2. **Clause Retention:** The program keeps too many deduced clauses (too many conclusions) in its database of information.
3. **Size of the Deduction Steps:** The inference rules do not take deduction steps (steps of reasoning) of the appropriate size.
4. **Inadequate Focus:** The program gets lost too easily.
5. **Metarules:** No adequate guidelines exist for selecting the appropriate representation, inference rule(s), strategy or strategies, transformations for canonicalization (*demodulators*), and type of information-discarding procedure (*subsumption*) to be employed. This obstacle focuses on guidelines for the effective use of an automated reasoning program.
6. **Database Indexing:** The program requires too much time to find the appropriate information in its database. This obstacle focuses on implementation.

Far more experimentation is required for the continued evolution of the clause language paradigm and for establishing a metatheory for its most effective use. For this chapter to be complete, it must contain material that permits one to begin or continue research, whether directed to theory, to implementation, or to application. Therefore, we focus in Section 6 on challenge problems for testing, comparing, and evaluating programs, approaches, and ideas. Some of the included problems remain open at the time of this writing, but we conjecture that these are amenable to attack with an existing automated reasoning program. Since a frequent request concerns research topics—suitable for a doctoral dissertation, for example—we present, in Section 7, basic research problems and discuss the current state of automated reasoning.

To further encourage and support research, we include in Section 8 information on appropriate books, on obtaining an automated reasoning program, and on obtaining access to a database of problems of various types. One can now easily obtain by anonymous FTP a portable program [McCune, 1990] that can assist in systematically seeking shorter proofs, identifying dependent axioms, formulating conjectures, checking proofs, finding new axiom systems, and (the familiar) proving theorems.