Manuel Kauers
Manfred Kerber
Robert Miner
Wolfgang Windsteiger (Eds.)

LNAI 4573

Towards Mechanized Mathematical Assistants

14th Symposium, Calculemus 2007
6th International Conference, MKM 2007
Hagenberg, Austria, June 2007, Proceedings

Springer

Manuel Kauers   Manfred Kerber
Robert Miner   Wolfgang Windsteiger (Eds.)

Towards Mechanized Mathematical Assistants

14th Symposium, Calculemus 2007
6th International Conference, MKM 2007
Hagenberg, Austria, June 27-30, 2007
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

Manuel Kauers
Wolfgang Windsteiger

Johannes Kepler University
Research Institute for Symbolic Computation, Linz, Austria
E-mail: {Manuel.Kauers, Wolfgang.Windsteiger}@risc.uni-linz.ac.at

Manfred Kerber
The University of Birmingham
School of Computer Science, Birmingham B15 2TT, England
E-mail: M.Kerber@cs.bham.ac.uk

Robert Miner
Design Science, Inc., St. Paul, Minnesota, USA
E-mail: robertm@dessci.com

Library of Congress Control Number: 2007928735

CR Subject Classification (1998):
I.2.2, I.2, H.3, H.2.8, I.7.2, F.4.1, H.4, C.2.4, G.4, I.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN	0302-9743
ISBN-10	3-540-73083-4 Springer Berlin Heidelberg New York
ISBN-13	978-3-540-73083-5 Springer Berlin Heidelberg New York

Springer is a part of Springer Science+Business Media

springer.com

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 12077151 06/3180 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 4573

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 4573: M. Kauers, M. Kerber, R. Miner, W. Windsteiger (Eds.), Towards Mechanized Mathematical Assistants. XIII, 407 pages. 2007.

Vol. 4539: N.H. Bshouty, C. Gentile (Eds.), Learning Theory. XII, 634 pages. 2007.

Vol. 4529: P. Melin, O. Castillo, L.T. Aguilar, J. Kacprzyk, W. Pedrycz (Eds.), Foundations of Fuzzy Logic and Soft Computing. XIX, 830 pages. 2007.

Vol. 4509: Z. Kobti, D. Wu (Eds.), Advances in Artificial Intelligence. XII, 552 pages. 2007.

Vol. 4496: N.T. Nguyen, A. Grzech, R.J. Howlett, L.C. Jain (Eds.), Agent and Multi-Agent Systems: Technologies and Applications. XXI, 1046 pages. 2007.

Vol. 4483: C. Baral, G. Brewka, J. Schlipf (Eds.), Logic Programming and Nonmonotonic Reasoning. IX, 327 pages. 2007.

Vol. 4482: A. An, J. Stefanowski, S. Ramanna, C.J. Butz, W. Pedrycz, G. Wang (Eds.), Rough Sets, Fuzzy Sets, Data Mining and Granular Computing. XIV, 585 pages. 2007.

Vol. 4481: J. Yao, P. Lingras, W.-Z. Wu, M. Szczuka, N.J. Cercone, D. Ślęzak (Eds.), Rough Sets and Knowledge Technology. XIV, 576 pages. 2007.

Vol. 4476: V. Gorodetsky, C. Zhang, V.A. Skormin, L. Cao (Eds.), Autonomous Intelligent Systems: Multi-Agents and Data Mining. XIII, 323 pages. 2007.

Vol. 4452: M. Fasli, O. Shehory (Eds.), Agent-Mediated Electronic Commerce. VIII, 249 pages. 2007.

Vol. 4451: T.S. Huang, A. Nijholt, M. Pantic, A. Pentland (Eds.), Artifical Intelligence for Human Computing. XVI, 359 pages. 2007.

Vol. 4438: L. Maicher, A. Sigel, L.M. Garshol (Eds.), Leveraging the Semantics of Topic Maps. X, 257 pages. 2007.

Vol. 4429: R. Lu, J.H. Siekmann, C. Ullrich (Eds.), Cognitive Systems. X, 161 pages. 2007.

Vol. 4426: Z.-H. Zhou, H. Li, Q. Yang (Eds.), Advances in Knowledge Discovery and Data Mining. XXV, 1161 pages. 2007.

Vol. 4411: R.H. Bordini, M. Dastani, J. Dix, A.E.F. Seghrouchni (Eds.), Programming Multi-Agent Systems. XIV, 249 pages. 2007.

Vol. 4410: A. Branco (Ed.), Anaphora: Analysis, Algorithms and Applications. X, 191 pages. 2007.

Vol. 4399: T. Kovacs, X. Llorà, K. Takadama, P.L. Lanzi, W. Stolzmann, S.W. Wilson (Eds.), Learning Classifier Systems. XII, 345 pages. 2007.

Vol. 4390: S.O. Kuznetsov, S. Schmidt (Eds.), Formal Concept Analysis. X, 329 pages. 2007.

Vol. 4389: D. Weyns, H.V.D. Parunak, F. Michel (Eds.), Environments for Multi-Agent Systems III. X, 273 pages. 2007.

Vol. 4384: T. Washio, K. Satoh, H. Takeda, A. Inokuchi (Eds.), New Frontiers in Artificial Intelligence. IX, 401 pages. 2007.

Vol. 4371: K. Inoue, K. Satoh, F. Toni (Eds.), Computational Logic in Multi-Agent Systems. X, 315 pages. 2007.

Vol. 4369: M. Umeda, A. Wolf, O. Bartenstein, U. Geske, D. Seipel, O. Takata (Eds.), Declarative Programming for Knowledge Management. X, 229 pages. 2006.

Vol. 4342: H. de Swart, E. Orłowska, G. Schmidt, M. Roubens (Eds.), Theory and Applications of Relational Structures as Knowledge Instruments II. X, 373 pages. 2006.

Vol. 4335: S.A. Brueckner, S. Hassas, M. Jelasity, D. Yamins (Eds.), Engineering Self-Organising Systems. XII, 212 pages. 2007.

Vol. 4334: B. Beckert, R. Hähnle, P.H. Schmitt (Eds.), Verification of Object-Oriented Software. XXIX, 658 pages. 2007.

Vol. 4333: U. Reimer, D. Karagiannis (Eds.), Practical Aspects of Knowledge Management. XII, 338 pages. 2006.

Vol. 4327: M. Baldoni, U. Endriss (Eds.), Declarative Agent Languages and Technologies IV. VIII, 257 pages. 2006.

Vol. 4314: C. Freksa, M. Kohlhase, K. Schill (Eds.), KI 2006: Advances in Artificial Intelligence. XII, 458 pages. 2007.

Vol. 4304: A. Sattar, B.-H. Kang (Eds.), AI 2006: Advances in Artificial Intelligence. XXVII, 1303 pages. 2006.

Vol. 4303: A. Hoffmann, B.-H. Kang, D. Richards, S. Tsumoto (Eds.), Advances in Knowledge Acquisition and Management. XI, 259 pages. 2006.

Vol. 4293: A. Gelbukh, C.A. Reyes-Garcia (Eds.), MICAI 2006: Advances in Artificial Intelligence. XXVIII, 1232 pages. 2006.

Vol. 4289: M. Ackermann, B. Berendt, M. Grobelnik, A. Hotho, D. Mladenič, G. Semeraro, M. Spiliopoulou, G. Stumme, V. Svátek, M. van Someren (Eds.), Semantics, Web and Mining. X, 197 pages. 2006.

Vol. 4285: Y. Matsumoto, R.W. Sproat, K.-F. Wong, M. Zhang (Eds.), Computer Processing of Oriental Languages. XVII, 544 pages. 2006.

Vol. 4274: Q. Huo, B. Ma, E.-S. Chng, H. Li (Eds.), Chinese Spoken Language Processing. XXIV, 805 pages. 2006.

Vol. 4265: L. Todorovski, N. Lavrač, K.P. Jantke (Eds.), Discovery Science. XIV, 384 pages. 2006.

Vol. 4264: J.L. Balcázar, P.M. Long, F. Stephan (Eds.), Algorithmic Learning Theory. XIII, 393 pages. 2006.

Vol. 4259: S. Greco, Y. Hata, S. Hirano, M. Inuiguchi, S. Miyamoto, H.S. Nguyen, R. Słowiński (Eds.), Rough Sets and Current Trends in Computing. XXII, 951 pages. 2006.

Vol. 4253: B. Gabrys, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part III. XXXII, 1301 pages. 2006.

Vol. 4252: B. Gabrys, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part II. XXXIII, 1335 pages. 2006.

Vol. 4251: B. Gabrys, R.J. Howlett, L.C. Jain (Eds.), Knowledge-Based Intelligent Information and Engineering Systems, Part I. LXVI, 1297 pages. 2006.

Vol. 4248: S. Staab, V. Svátek (Eds.), Managing Knowledge in a World of Networks. XIV, 400 pages. 2006.

Vol. 4246: M. Hermann, A. Voronkov (Eds.), Logic for Programming, Artificial Intelligence, and Reasoning. XIII, 588 pages. 2006.

Vol. 4223: L. Wang, L. Jiao, G. Shi, X. Li, J. Liu (Eds.), Fuzzy Systems and Knowledge Discovery. XXVIII, 1335 pages. 2006.

Vol. 4213: J. Fürnkranz, T. Scheffer, M. Spiliopoulou (Eds.), Knowledge Discovery in Databases: PKDD 2006. XXII, 660 pages. 2006.

Vol. 4212: J. Fürnkranz, T. Scheffer, M. Spiliopoulou (Eds.), Machine Learning: ECML 2006. XXIII, 851 pages. 2006.

Vol. 4211: P. Vogt, Y. Sugita, E. Tuci, C.L. Nehaniv (Eds.), Symbol Grounding and Beyond. VIII, 237 pages. 2006.

Vol. 4203: F. Esposito, Z.W. Raś, D. Malerba, G. Semeraro (Eds.), Foundations of Intelligent Systems. XVIII, 767 pages. 2006.

Vol. 4201: Y. Sakakibara, S. Kobayashi, K. Sato, T. Nishino, E. Tomita (Eds.), Grammatical Inference: Algorithms and Applications. XII, 359 pages. 2006.

Vol. 4200: I.F.C. Smith (Ed.), Intelligent Computing in Engineering and Architecture. XIII, 692 pages. 2006.

Vol. 4198: O. Nasraoui, O. Zaïane, M. Spiliopoulou, B. Mobasher, B. Masand, P.S. Yu (Eds.), Advances in Web Mining and Web Usage Analysis. IX, 177 pages. 2006.

Vol. 4196: K. Fischer, I.J. Timm, E. André, N. Zhong (Eds.), Multiagent System Technologies. X, 185 pages. 2006.

Vol. 4188: P. Sojka, I. Kopeček, K. Pala (Eds.), Text, Speech and Dialogue. XV, 721 pages. 2006.

Vol. 4183: J. Euzenat, J. Domingue (Eds.), Artificial Intelligence: Methodology, Systems, and Applications. XIII, 291 pages. 2006.

Vol. 4180: M. Kohlhase, OMDoc – An Open Markup Format for Mathematical Documents [version 1.2]. XIX, 428 pages. 2006.

Vol. 4177: R. Marín, E. Onaindía, A. Bugarín, J. Santos (Eds.), Current Topics in Artificial Intelligence. XV, 482 pages. 2006.

Vol. 4160: M. Fisher, W. van der Hoek, B. Konev, A. Lisitsa (Eds.), Logics in Artificial Intelligence. XII, 516 pages. 2006.

Vol. 4155: O. Stock, M. Schaerf (Eds.), Reasoning, Action and Interaction in AI Theories and Systems. XVIII, 343 pages. 2006.

Vol. 4149: M. Klusch, M. Rovatsos, T.R. Payne (Eds.), Cooperative Information Agents X. XII, 477 pages. 2006.

Vol. 4140: J.S. Sichman, H. Coelho, S.O. Rezende (Eds.), Advances in Artificial Intelligence - IBERAMIA-SBIA 2006. XXIII, 635 pages. 2006.

Vol. 4139: T. Salakoski, F. Ginter, S. Pyysalo, T. Pahikkala (Eds.), Advances in Natural Language Processing. XVI, 771 pages. 2006.

Vol. 4133: J. Gratch, M. Young, R. Aylett, D. Ballin, P. Olivier (Eds.), Intelligent Virtual Agents. XIV, 472 pages. 2006.

Vol. 4130: U. Furbach, N. Shankar (Eds.), Automated Reasoning. XV, 680 pages. 2006.

Vol. 4120: J. Calmet, T. Ida, D. Wang (Eds.), Artificial Intelligence and Symbolic Computation. XIII, 269 pages. 2006.

Vol. 4118: Z. Despotovic, S. Joseph, C. Sartori (Eds.), Agents and Peer-to-Peer Computing. XIV, 173 pages. 2006.

Vol. 4114: D.-S. Huang, K. Li, G.W. Irwin (Eds.), Computational Intelligence, Part II. XXVII, 1337 pages. 2006.

Vol. 4108: J.M. Borwein, W.M. Farmer (Eds.), Mathematical Knowledge Management. VIII, 295 pages. 2006.

Vol. 4106: T.R. Roth-Berghofer, M.H. Göker, H.A. Güvenir (Eds.), Advances in Case-Based Reasoning. XIV, 566 pages. 2006.

Vol. 4099: Q. Yang, G. Webb (Eds.), PRICAI 2006: Trends in Artificial Intelligence. XXVIII, 1263 pages. 2006.

Vol. 4095: S. Nolfi, G. Baldassarre, R. Calabretta, J.C.T. Hallam, D. Marocco, J.-A. Meyer, O. Miglino, D. Parisi (Eds.), From Animals to Animats 9. XV, 869 pages. 2006.

Vol. 4093: X. Li, O.R. Zaïane, Z. Li (Eds.), Advanced Data Mining and Applications. XXI, 1110 pages. 2006.

Vol. 4092: J. Lang, F. Lin, J. Wang (Eds.), Knowledge Science, Engineering and Management. XV, 664 pages. 2006.

Vol. 4088: Z.-Z. Shi, R. Sadananda (Eds.), Agent Computing and Multi-Agent Systems. XVII, 827 pages. 2006.

Vol. 4087: F. Schwenker, S. Marinai (Eds.), Artificial Neural Networks in Pattern Recognition. IX, 299 pages. 2006.

Vol. 4068: H. Schärfe, P. Hitzler, P. Øhrstrøm (Eds.), Conceptual Structures: Inspiration and Application. XI, 455 pages. 2006.

Vol. 4065: P. Perner (Ed.), Advances in Data Mining. XI, 592 pages. 2006.

Vol. 4062: G.-Y. Wang, J.F. Peters, A. Skowron, Y. Yao (Eds.), Rough Sets and Knowledge Technology. XX, 810 pages. 2006.

Preface

This volume contains the collected contributions of two conferences, Calculemus 2007 and MKM 2007. Calculemus 2007 was the 14th in a series of conferences dedicated to the integration of computer algebra systems (CAS) and automated deduction systems (ADS). MKM 2007 was the sixth International Conference on Mathematical Knowledge Management, an emerging interdisciplinary field of research in the intersection of mathematics, computer science, library science, and scientific publishing. Both conferences aimed to provide mechanized mathematical assistants.

Although the two conferences have separate communities and separate foci, there is a significant overlap in the interests in building mechanized mathematical assistants. For this reason it was decided to collocate the two events in 2007 for the first time, at RISC in Hagenberg, Austria. The number and quality of the submissions show that this was a good decision. While the proceedings are shared, the submission process was separate. The responsibility for acceptance/rejection rests completely with the two separate Program Committees.

By this collocation we made a contribution against the fragmentation of communities which work on different aspects of different independent branches, traditional branches (e.g., computer algebra and theorem proving), as well as newly emerging ones (on user interfaces, knowledge management, theory exploration, etc.). This will also facilitate the development of integrated mechanized mathematical assistants that will be routinely used by mathematicians, computer scientists, and engineers in their every-day business.

In total, 23 papers were submitted to Calculemus. For each paper there were three reviews and, finally, ten papers were accepted for publication in these proceedings. MKM received 52 submissions (more than double last year's number). For each paper there were at least two reviews; if the evaluation was not uniform we had three and in some cases four reviews. After discussions, we accepted 19 high-quality papers for these proceedings. In the preparation of these proceedings and in managing the whole discussion process, Andrei Voronkov's EasyChair conference management system proved itself an excellent tool. In addition to the contributed papers, abstracts of the invited speakers of MKM are found in these proceedings.

April 2007

Manuel Kauers
Manfred Kerber
Robert Miner
Wolfgang Windsteiger

Calculemus & MKM Organization

Conference Chair	Wolfgang Windsteiger (RISC, Linz, Austria)
Local Arrangements	Laura Kovács (RISC, Linz, Austria)
WWW:	http://www.risc.uni-linz.ac.at/about/conferences/summer2007/

Sponsors

Calculemus and MKM greatfully acknowledge the financial support of the following institutions:

- Bundesministerium für Wissenschaft und Bildung, Österreich
  (Federal Minister of Science and Education, Austria)
- Land Oberösterreich (Upper Austrian Government)
- Raiffaisen Landesbank Oberösterreich
- Johannes Kepler University
- Linzer Hochschulfond
- Spezialforschungsbereich SFB F013
  "Numerical and Symbolic Scientific Computing"
- Research Institute for Symbolic Computation (RISC)
- RISC Software GmbH
- uni software plus, Mathematica reseller, Austria

Calculemus 2007 Organization

Program Committee

Alessandro Armando	University of Genova, Italy
Christoph Benzmüller	University of Cambridge, UK
Olga Caprotti	University of Helsinki, Finland
Jacques Carette	McMaster University, Canada
Timothy Daly	Carnegie Mellon University, USA
William Farmer	McMaster University, Canada
Keith Geddes	University of Waterloo, Canada
Tom Hales	University of Pittsburgh, USA
Hoon Hong	North Carolina State University, USA
Deepak Kapur	University of New Mexico, USA
Manuel Kauers	RISC-Linz, Austria (*Co-chair*)
Laura Kovacs	RISC-Linz, Austria
Petr Lisonek	Simon Fraser University, Canada
Renaud Rioboo	Universtite Pierre et Marie Curie, France

Volker Sorge	University of Birmingham, UK
Thomas Sturm	Carnegie Mellon University, USA
Klaus Sutner	University of Passau, Germany
Wolfgang Windsteiger	RISC-Linz, Austria (*Co-chair*)

External Reviewers

Chad Brown	Martin Kreuzer	Wolfgang Schreiner
Stephen Forrest	Aless Lasaruk	Dongming Wang
Therese Hardin	Piotr Rudnicki	Tetsu Yamaguchi

MKM 2007 Organization

Program Committee

Andrea Asperti	University of Bologna, Italy
Laurent Bernardin	Maplesoft, Canada
Jonathan Borwein	Dalhousie University, Halifax, Canada
Thierry Bouche	Université de Grenoble I, France
Bruno Buchberger	Johannes Kepler University, Linz, Austria
Paul Cairns	University College London, UK
Olga Caprotti	University of Helsinki, Finland
Bruce Char	Drexel University, Philadelphia, USA
Simon Colton	Imperial College, London, UK
Mike Dewar	Numerical Algorithms Group, Oxford, UK
William Farmer	McMaster University, Hamilton, Canada
Herman Geuvers	Radboud Univ. Nijmegen, The Netherlands
Tetsuo Ida	University of Tsukuba, Japan
Mateja Jamnik	University of Cambridge, UK
Fairouz Kamareddine	Heriot-Watt University, UK
Manfred Kerber	University of Birmingham, UK (*Co-chair*)
Michael Kohlhase	International University Bremen, Germany
Paul Libbrecht	DFKI Saarbrücken, Germany
Robert Miner	Design Science, Inc., USA (*Co-chair*)
Bengt Nordström	Chalmers University of Technology, Göteborg, Sweden
Ross Reedstrom	Rice University, USA
Eugénio Rocha	University of Aveiro, Portugal
Alan Sexton	University of Birmingham, UK
Andrzej Trybulec	University of Białystok, Poland
Stephen Watt	The University of Western Ontario, Canada
Abdou Youssef	George Washington University, Washington, DC, USA

External Reviewers

Pierre Corbineau
Cezary Kaliszyk
Pouya Larjani
Mircea Marin
Russell O'Connor
Florian Rabe
Freek Wiedijk
Claudio Sacerdoti Coen
Robert Lamar
Lionel Mamane
Normen Mueller
Martijn Oostdijk
Krzysztof Retel
Jian Xu
Jeremy Gow
Christoph Lange
Manuel Maarek
Christine Mueller
Matti Pauna
Clare So

Table of Contents

Contributions to Calculemus 2007

Contributions to MKM 2007

Executing in Common Lisp, Proving in ACL2*

Mirian Andrés, Laureano Lambán, and Julio Rubio

Departamento de Matemáticas y Computación, Universidad de La Rioja, Edificio Vives. Calle Luis de Ulloa s/n, E-26004 Logroño, Spain
{mirian.andres,lalamban,julio.rubio}@unirioja.es

Abstract. In this paper, an approach to integrate an already-written Common Lisp program for algebraic manipulation with ACL2 proofs of properties of that program is presented. We report on a particular property called "cancellation theorem", which has been proved in ACL2, and could be applied to several problems in the field of Computational Algebraic Topology.

1 Introduction

Kenzo is a Common Lisp program [10] designed by Sergeraert, implementing his ideas on *Constructive Algebraic Topology* [19]. Kenzo, and its predecessor EAT [21], were capable of computing homology groups unknown by any other means. Kenzo continues to evolve and has been recently released as an open source computer algebra system [10] and extended with new modules on Koszul Homology [20], Spectral Sequences [18] and Coalgebras [4].

Several years ago a project was launched to analyze the Kenzo system by means of formal methods. The objective of the project is twofold. Better knowledge of the internal processes and structures in Kenzo is intented, thus increasing the reliability of the system. Besides, Kenzo is also a good "laboratory" (due to its structural richness and to the presence of challenging results which have been obtained using it) to experiment with different tools and approaches in the field of formal methods in Software Engineering, allowing the analyst to compare them, to evaluate them and, hopefully, to apply them to other fields unrelated to Algebraic Topology or Computer Algebra.

The first efforts were devoted to the Algebraic Specification of EAT [13] and Kenzo [8,9]. After that, these rather theoretical results were put into practice through *theorem provers*. The tactical assistant Isabelle [17] was chosen for the first studies [1,2] on the application of automated theorem proving in the area of Algebraic Topology. These preliminary works led to the recent Isabelle mechanized proof of the Basic Perturbation Lemma [3], one of the central results in Algorithmic Homological Algebra. Other lines of research include modeling and proving with Coq [5], and programming and proving with the system FoCaL [6].

In this paper we report on a relative approach, by using the theorem prover ACL2 [12]. The limitations of this prover with respect to Isabelle or Coq are

* Partially supported by Comunidad Autónoma de La Rioja, project ANGI-2005/19, and by Ministerio de Educación y Ciencia, project MTM2006-06513.

M. Kauers et al. (Eds.): MKM/Calculemus 2007, LNAI 4573, pp. 1–12, 2007.

well-known and are essentially related to the underlying logics. ACL2 is based on a weak form of first order logic, while both Coq and Isabelle can work with higher order logic. On the positive side, ACL2 is based on Common Lisp (as Kenzo itself) and is very suitable when linking proofs and running programs. In addition, the treatment of Symbolic Computation problems with the help of ACL2 has obtained important successes in recent years (see, for instance, [15]).

The organization of the paper is as follows. The next section introduces our methodological approach to relate an already-written program with the proofs of properties in ACL2. Section 3 and 4 are devoted to introduce, respectively, our motivating examples from Homological Algebra and the basic data structures and proofs in ACL2. Section 5 presents the main contribution of the paper, reporting on the automated proof of a "cancellation theorem". This theorem is applied in Section 6 to the proof of an algebraic property of our programs. The paper ends with the section of conclusions and future work, and the bibliography.

2 Proving and Then... Testing

There are many ways in which Symbolic Computation (or programming, more generally) can interplay with theorem proving. For instance, Computer Algebra programs can be used as oracles for theorem provers. In the other direction, theorem provers can be used to ensure the correctness of Computer Algebra programs. In this paper we will introduce a third manner of interaction: theorem provers can be used for automated-testing of programs. Although it is usually considered that testing is easier than proving, and so that testing should occur in early stages of the quality control cycle, our proposal is the reversal (in a sense which will be clear later on): *first proving and then... testing.* Of course, the complete picture of our view is more complex than indicated by that simplistic phrase. Let us explore it in a concrete situation.

Let us assume that someone gave us a Common Lisp `program1` with the following characteristics:

- it is difficult to test, perhaps because it produces results difficult to interpret, or, even worse, some of its results are unknown by any other means, and
- the program correctness is difficult to prove, perhaps due to being logically complex, based on higher-order constructions, for instance.

An example of such a `program1` could be the Kenzo system, which has been developed in Common Lisp and has been successfully tested for more than fifteen years, but ... not always: some of the results found with the help of Kenzo continue to be unverifiable by any other means at this moment (homology groups of some iterated loop spaces, for instance; see [10]). In addition, Kenzo is based on both object-orientation and higher-order functional programming, in such a way that its formal specification is challenging (see [13,8,9]), and therefore its verification with theorem provers poses problems far from trivial. The formal specification and verification of some of the `algorithms` appearing in Kenzo have been carried out with the Isabelle assistant [17], and were explained in

[1] and [2]. The most relevant result in this line is the recent Aransay's proof in Isabelle/HOL of the BPL, the Basic Perturbation Lemma [3]. The BPL is one of the most important theorems and algorithms used to build Kenzo. But, independently of the merits of this mechanized proof of the BPL, the distance with respect to the *programs* implementing the BPL in Kenzo, continues to be quite large.

Since our goal is to verify *real* Common Lisp programs, a sensible idea should be to use the ACL2 system to devise proofs (instead of Isabelle or Coq). ACL2 [12] is both a programming language and an environment to produce proofs of properties of programs. As programming language, ACL2 is an extension of a sub-language of Common Lisp. The extensions added to Common Lisp in ACL2 are not relevant for our work. On the contrary, the features erased from Common Lisp in ACL2 are very important with respect to Kenzo. In particular, ACL2 does not allow the programmer to use higher-order functionals, a tool intensively employed in Kenzo. Thus, in order to study a Common Lisp `program1` within ACL2, we are proposing to write a new Common Lisp `program2` emulating the behavior of `program1`, but programmed this time in ACL2.

Let us enumerate the characteristics of this situation:

- `program1` is
  - already written
  - in Common Lisp (not necessarily in ACL2);
  - efficient;
  - tested;
  - unproved.
- `program2` is
  - specially designed to be proved;
  - programmed in ACL2 (and Common Lisp);
  - efficient or not: irrelevant;
  - tested;
  - proved in ACL2.

In our approach, `program2` is *supposed to be equivalent* to `program1`. But we do not pretend to prove this equivalence: this option would lead us to a form of ill-founded recursion. Our aim should be to use the *highly reliable* `program2` to perform automated testing of the *efficient* `program1`.

The following toy program will illustrate this idea:

```
(defun automated-testing ()
   (let ((case (generate-test-case)))
     (if (not (equal (program1 case)
                     (program2 case)))
         (report-on-failure case))))
```

Note that it is an (unverified!) Common Lisp program, but not an ACL2 one (at least, if `program1` is not).

The relationship of these ideas with *Model Checking* is appealing. Even if the field of application (reactive systems modeled as state machines) and the formal methods used (temporal logics) are different from ours, at least in the standard literature on Model Checking [7], the underlying philosophy is the same. In our case, the system (an already written `program1`) is abstracted into a model (`program2`). Then, formal methods (theorem proving in our case) are used to get theoretical properties of the model (the correctness of `program2`, proved in ACL2). The final step is to interpret the results obtained from the model with respect to reality (automated testing of the `program1` against `program2`).

As in Model Checking, one of the important bottlenecks of the method is to build a model which is an accurate representation of the system to be modeled. In Model Checking one such difficult step occurs when an infinite system (that is to say, a system with an unbounded number of possible reachable states) is modeled by means of a finite graph (the condition of finiteness is mandatory, because the checking of properties is done by exhaustive traversal of state spaces).

In our context, it is hopeless to apply our method to the whole Kenzo system. The most important constraint is that we must restrict our ACL2 study to the parts of Kenzo which are *first-order*[1]. This excludes large (and interesting!) fragments of Kenzo, that should be analyzed by using tools such as Isabelle (as in [1], [2] or [3]) or Coq.

Once a part of Kenzo with this characteristic has been chosen (let us call it `program1`), the (heuristic) transformations we apply to construct the model `program2` are the following:

- iterations and loops are replaced by recursive functions (this step could be automated);
- first-order functional programming is replaced by standard functions[2];
- data structures are "flattened" to lists: objects, structs and arrays are replaced by convenient nested lists;
- destructive operations are replaced by the corresponding constructive ones (this is a problematic point, but destructive updates appear in very precisely located Kenzo fragments, and so this task is quite relaxed).

With these cautions, it is hoped that `program2` accurately models `program1`, and then our strategy could be safely applied.

3 Homological Algebra

A first application of the ideas presented in the previous section arises from two different on-going projects devoted to analyze formally Kenzo [10], the system for computing in Algebraic Topology.

1 Interestingly enough, this constraint seems related, in some sense, with the finite/infinite dichotomy evoked previously on Model Checking.

2 For instance, an occurrence of `(mapcar #'cadr l)` should be replaced by `(mapcadar l)` where the new function `mapcadar` is simply:

```
(defun mapcadar (l) (if (endp l) l (cons (cadar l) (mapcadar (cdr l)))))
```