Julian Miller Marco Tomassini
Pier Luca Lanzi Conor Ryan
Andrea G. B. Tettamanzi
William B. Langdon (Eds.)

LNCS 2038

Genetic Programming

4th European Conference, EuroGP 2001
Lake Como, Italy, April 2001
Proceedings

Springer

Julian Miller Marco Tomassini
Pier Luca Lanzi Conor Ryan
Andrea G.B. Tettamanzi
William B. Langdon (Eds.)

Genetic Programming

4th European Conference, EuroGP 2001
Lake Como, Italy, April 18-20, 2001
Proceedings

Springer

Series Editors

Gerhard Goos, Karlsruhe University, Germany
Juris Hartmanis, Cornell University, NY, USA
Jan van Leeuwen, Utrecht University, The Netherlands

Volume Editors

Julian Miller
University of Birmingham, School of Computer Science
E-mail: j.miller@cs.bham.ac.uk

Marco Tomassini
University of Lausanne, Computer Science Institute
E-mail: marco.tomassini@iismail.unil.ch

Pier Luca Lanzi
Politecnico di Milano, Dipartimento di Elettronica e Informazione
E-mail: lanzi@morgana.elet.polimi.it

Conor Ryan
University of Limerick, Computer Science and Information Systems
E-mail: Conor.Ryan@ul.ie

Andrea G.B. Tettamanzi
Università degli Studi di Milano, Dipartimento di Tecnologie dell'Informazione
E-mail: andrea.tettamanzi@unimi.it

William B. Langdon
University College London, Department of Computer Science
E-mail: W.Langdon@cs.ucl.ac.uk

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Genetic programming : 4th European conference ; proceedings / EuroGP 2001, Lake Como, Italy, UK, April 18 - 20, 2001. Julian Miller ... (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Hong Kong ; London ; Milan ; Paris ; Singapore ; Tokyo : Springer, 2001
(Lecture notes in computer science ; Vol. 2038)
ISBN 3-540-41899-7

CR Subject Classification (1998): D.1, F.1, F.2, I.5, I.2, J.3

ISSN 0302-9743
ISBN 3-540-41899-7 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Printed in Germany

Typesetting: Camera-ready by author, data conversion by Christian Grosche, Hamburg
Printed on acid-free paper SPIN: 10782492 06/3142 5 4 3 2 1 0

Lecture Notes in Computer Science 2038
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Preface

In this volume are the proceedings of the fourth European conference on Genetic Programming (EuroGP 2001) which took place at Lake Como in Italy on April, 18–20 2001. EuroGP has become firmly established as the premier European event devoted to Genetic Programming. EuroGP began life in 1998 as an international workshop and was held in Paris (14–15 April, LNCS 1391). After that it was held in Göteborg, Sweden (26–27 May 1999, LNCS 1598). Its first appearance as a conference was last year in Edinburgh in Scotland (15–16 April, LNCS 1802). Each year EuroGP has been co-located with a series of specialist workshops (LNCS 1468, 1596, 1803). This year was no exception and EvoWorkshops 2001 were also held at Lake Como (18-19 April, LNCS 2037).

Genetic Programming (GP) refers to a branch of Evolutionary Computation in which computer programs are automatically generated over a period of time using a process that mimics Darwinian evolution. The 30 papers in these proceedings more than amply demonstrate the wide and varied applicability of GP. There are papers that apply GP to robotics, artificial retina, character recognition, financial prediction, digital filter and electronic circuit design, image processing, data fusion, and biosequencing. In addition there are many papers that address foundational and theoretical issues.

A rigorous double-blind refereeing system was applied to the 42 submitted papers. This resulted in 17 plenary talks (40% of those submitted) and 13 research posters. Every submitted paper was reviewed by a minimum of two members of the International Program Committee, and if there was disagreement by a third reviewer. The Program Committee was carefully selected for their knowledge and expertise, and, as far as possible, papers were matched with the reviewers' particular interests and specialist expertise. The results of this process are seen here in the high quality of papers published within this volume. Many of these are by internationally recognised researchers.

The 30 published papers came from many European countries with a noticeable proportion from the Americas.

We would like to express our sincere thanks especially to the two internationally renowned invited speakers who gave keynote talks at the conference: Professor Enrico Coen of the John Innes Centre, UK and Professor Lee Altenberg of the University of Hawaii at Manoa. Professor Coen's talk was also shared with EvoWorkshops 2001. We would also like to thank Dr. Riccardo Poli of the School of Computer Science at the University of Birmingham for kindly agreeing to give a tutorial on GP.

This conference would have been considerably poorer without the support of many people. Firstly we would like to thank the very busy members of the Program Committee for their diligence, patience, and dedication in the task of providing high quality reviews. We would also like to thank EvoNET, the Network of Excellence in Evolutionary Computing, for their support, in particular, Jennifer Willies and Chris Osborne for their help, especially their sterling work on the registration and the conference web site. Thanks also to Chris Osborne and Mij Kelly for their assistance with the production of the conference poster. Finally we would like to thank the members of EvoGP, the EvoNET working group on Genetic Programming.

April 2001 Julian Miller, Marco Tomassini, Pier Luca Lanzi, Conor Ryan, Andrea G.B. Tettamanzi, William B. Langdon

Organisation

EuroGP 2001 was organised by EvoGP, the EvoNet Working Group on Genetic Programming.

Organising Committee

Program co-chair:	Julian Miller (University of Birmingham, UK)
Program co-chair:	Marco Tomassini (University of Lausanne, Switzerland)
Publicity chair:	Conor Ryan (University of Limerick, Ireland)
Local co-chairs:	Pier Luca Lanzi (Politecnico di Milano, Italy)
	Andrea G.B. Tettamanzi (University of Milan, Italy)
Publication co-chair:	William B. Langdon (University College, London, UK)

Program Committee

Wolfgang Banzhaf, University of Dortmund, Germany
Forest Bennett III, FX Palo Alto Laboratory, USA
Shu-Heng Chen, National Chengchi University, Taiwan
Marco Dorigo, Free University of Brussels, Belgium
Terry Fogarty, South Bank University, UK
James A. Foster, University of Idaho, USA
Hitoshi Iba, University of Tokyo, Japan
Christian Jacob, University of Calgary, Canada
Maarten Keijzer, Danish Hydraulics Insitute, Denmark
Ibrahim Kuscu , University of Surrey , UK
William B. Langdon, University College London, UK
Sean Luke, University of Maryland, USA
Evelyne Lutton, INRIA, France
Nic McPhee, University of Minnesota, Morris, USA
Jean-Arcady Meyer, Université Pierre et Marie Curie, France
Julian Miller, University of Birmingham, UK
Peter Nordin, Chalmers University of Technology, Sweden
Simon Perkins, Los Alamos National Laboratories, USA
Riccardo Poli, University of Birmingham, UK
Joao C.F. Pujol, Centro de Desenvolvimento da Energia Nuclea, Brazil
Kazuhiro Saitou, University of Michigan, USA
Jonathan Rowe, University of Birmingham, UK
Peter Ross, Napier University, UK
Conor Ryan, University of Limerick, Ireland
Marc Schoenauer, Ecole Polytechnique, France
Moshe Sipper, EPFL, Switzerland

Michele Sebag, Ecole Polytechnique, France
Terry Soule, St Cloud State University, Minnesota, USA
Andrea G. B. Tettamanzi, Genetica - Advanced Software Architectures, Italy
Adrian Thompson, University of Sussex, UK
Marco Tomassini, Université de Lausanne, Switzerland
Hans-Michael Voigt, Center for Applied Computer Science, Berlin, German
Peter A. Whigham, University of Otago, New Zealand
Xin Yao, University of Birmingham, UK
Tina Yu, Chevron Information Technology Company, USA

Sponsoring Institutions

EvoNet: The Network of Excellence in Evolutionary Computing.

Lecture Notes in Computer Science

For information about Vols. 1–1933
please contact your bookseller or Springer-Verlag

Vol. 1934: J.S. White (Ed.), Envisioning Machine Translation in the Information Future. Proceedings, 2000. XV, 254 pages. 2000. (Subseries LNAI).

Vol. 1935: S.L. Delp, A.M. DiGioia, B. Jaramaz (Eds.), Medical Image Computing and Computer-Assisted Intervention – MICCAI 2000. Proceedings, 2000. XXV, 1250 pages. 2000.

Vol. 1936: P. Robertson, H. Shrobe, R. Laddaga (Eds.), Self-Adaptive Software. Proceedings, 2000. VIII, 249 pages. 2001.

Vol. 1937: R. Dieng, O. Corby (Eds.), Knowledge Engineering and Knowledge Management. Proceedings, 2000. XIII, 457 pages. 2000. (Subseries LNAI).

Vol. 1938: S. Rao, K.I. Sletta (Eds.), Next Generation Networks. Proceedings, 2000. XI, 392 pages. 2000.

Vol. 1939: A. Evans, S. Kent, B. Selic (Eds.), «UML» – The Unified Modeling Language. Proceedings, 2000. XIV, 572 pages. 2000.

Vol. 1940: M. Valero, K. Joe, M. Kitsuregawa, H. Tanaka (Eds.), High Performance Computing. Proceedings, 2000. XV, 595 pages. 2000.

Vol. 1941: A.K. Chhabra, D. Dori (Eds.), Graphics Recognition. Proceedings, 1999. XI, 346 pages. 2000.

Vol. 1942: H. Yasuda (Ed.), Active Networks. Proceedings, 2000. XI, 424 pages. 2000.

Vol. 1943: F. Koornneef, M. van der Meulen (Eds.), Computer Safety, Reliability and Security. Proceedings, 2000. X, 432 pages. 2000.

Vol. 1944: K.R. Dittrich, G. Guerrini, I. Merlo, M. Oliva, M.E. Rodriguez (Eds.), Objects and Databases. Proceedings, 2000. X, 199 pages. 2001.

Vol. 1945: W. Grieskamp, T. Santen, B. Stoddart (Eds.), Integrated Formal Methods. Proceedings, 2000. X, 441 pages. 2000.

Vol. 1946: P. Palanque, F. Paternò (Eds.), Interactive Systems. Proceedings, 2000. X, 251 pages. 2001.

Vol. 1947: T. Sørevik, F. Manne, R. Moe, A.H. Gebremedhin (Eds.), Applied Parallel Computing. Proceedings, 2000. XII, 400 pages. 2001.

Vol. 1948: T. Tan, Y. Shi, W. Gao (Eds.), Advances in Multimodal Interfaces – ICMI 2000. Proceedings, 2000. XVI, 678 pages. 2000.

Vol. 1949: R. Connor, A. Mendelzon (Eds.), Research Issues in Structured and Semistructured Database Programming. Proceedings, 1999. XII, 325 pages. 2000.

Vol. 1950: D. van Melkebeek, Randomness and Completeness in Computational Complexity. XV, 196 pages. 2000.

Vol. 1951: F. van der Linden (Ed.), Software Architectures for Product Families. Proceedings, 2000. VIII, 255 pages. 2000.

Vol. 1952: M.C. Monard, J. Simão Sichman (Eds.), Advances in Artificial Intelligence. Proceedings, 2000. XV, 498 pages. 2000. (Subseries LNAI).

Vol. 1953: G. Borgefors, I. Nyström, G. Sanniti di Baja (Eds.), Discrete Geometry for Computer Imagery. Proceedings, 2000. XI, 544 pages. 2000.

Vol. 1954: W.A. Hunt, Jr., S.D. Johnson (Eds.), Formal Methods in Computer-Aided Design. Proceedings, 2000. XI, 539 pages. 2000.

Vol. 1955: M. Parigot, A. Voronkov (Eds.), Logic for Programming and Automated Reasoning. Proceedings, 2000. XIII, 487 pages. 2000. (Subseries LNAI).

Vol. 1956: T. Coquand, P. Dybjer, B. Nordström, J. Smith (Eds.), Types for Proofs and Programs. Proceedings, 1999. VII, 195 pages. 2000.

Vol. 1957: P. Ciancarini, M. Wooldridge (Eds.), Agent-Oriented Software Engineering. Proceedings, 2000. X, 323 pages. 2001.

Vol. 1960: A. Ambler, S.B. Calo, G. Kar (Eds.), Services Management in Intelligent Networks. Proceedings, 2000. X, 259 pages. 2000.

Vol. 1961: J. He, M. Sato (Eds.), Advances in Computing Science – ASIAN 2000. Proceedings, 2000. X, 299 pages. 2000.

Vol. 1963: V. Hlaváč, K.G. Jeffery, J. Wiedermann (Eds.), SOFSEM 2000: Theory and Practice of Informatics. Proceedings, 2000. XI, 460 pages. 2000.

Vol. 1964: J. Malenfant, S. Moisan, A. Moreira (Eds.), Object-Oriented Technology. Proceedings, 2000. XI, 309 pages. 2000.

Vol. 1965: Ç. K. Koç, C. Paar (Eds.), Cryptographic Hardware and Embedded Systems – CHES 2000. Proceedings, 2000. XI, 355 pages. 2000.

Vol. 1966: S. Bhalla (Ed.), Databases in Networked Information Systems. Proceedings, 2000. VIII, 247 pages. 2000.

Vol. 1967: S. Arikawa, S. Morishita (Eds.), Discovery Science. Proceedings, 2000. XII, 332 pages. 2000. (Subseries LNAI).

Vol. 1968: H. Arimura, S. Jain, A. Sharma (Eds.), Algorithmic Learning Theory. Proceedings, 2000. XI, 335 pages. 2000. (Subseries LNAI).

Vol. 1969: D.T. Lee, S.-H. Teng (Eds.), Algorithms and Computation. Proceedings, 2000. XIV, 578 pages. 2000.

Vol. 1970: M. Valero, V.K. Prasanna, S. Vajapeyam (Eds.), High Performance Computing – HiPC 2000. Proceedings, 2000. XVIII, 568 pages. 2000.

Vol. 1971: R. Buyya, M. Baker (Eds.), Grid Computing – GRID 2000. Proceedings, 2000. XIV, 229 pages. 2000.

Vol. 1972: A. Omicini, R. Tolksdorf, F. Zambonelli (Eds.), Engineering Societies in the Agents World. Proceedings, 2000. IX, 143 pages. 2000. (Subseries LNAI).

Vol. 1973: J. Van den Bussche, V. Vianu (Eds.), Database Theory – ICDT 2001. Proceedings, 2001. X, 451 pages. 2001.

Vol. 1974: S. Kapoor, S. Prasad (Eds.), FST TCS 2000: Foundations of Software Technology and Theoretical Computer Science. Proceedings, 2000. XIII, 532 pages. 2000.

Vol. 1975: J. Pieprzyk, E. Okamoto, J. Seberry (Eds.), Information Security. Proceedings, 2000. X, 323 pages. 2000.

Vol. 1976: T. Okamoto (Ed.), Advances in Cryptology – ASIACRYPT 2000. Proceedings, 2000. XII, 630 pages. 2000.

Vol. 1977: B. Roy, E. Okamoto (Eds.), Progress in Cryptology – INDOCRYPT 2000. Proceedings, 2000. X, 295 pages. 2000.

Vol. 1978: B. Schneier (Ed.), Fast Software Encryption. Proceedings, 2000. VIII, 315 pages. 2001.

Vol. 1979: S. Moss, P. Davidsson (Eds.), Multi-Agent-Based Simulation. Proceedings, 2000. VIII, 267 pages. 2001. (Subseries LNAI).

Vol. 1983: K.S. Leung, L.-W. Chan, H. Meng (Eds.), Intelligent Data Engineering and Automated Learning – IDEAL 2000. Proceedings, 2000. XVI, 573 pages. 2000.

Vol. 1984: J. Marks (Ed.), Graph Drawing. Proceedings, 2001. XII, 419 pages. 2001.

Vol. 1985: J. Davidson, S.L. Min (Eds.), Languages, Compilers, and Tools for Embedded Systems. Proceedings, 2000. VIII, 221 pages. 2001.

Vol. 1987: K.-L. Tan, M.J. Franklin, J. C.-S. Lui (Eds.), Mobile Data Management. Proceedings, 2001. XIII, 289 pages. 2001.

Vol. 1988: L. Vulkov, J. Waśniewski, P. Yalamov (Eds.), Numerical Analysis and Its Applications. Proceedings, 2000. XIII, 782 pages. 2001.

Vol. 1989: M. Ajmone Marsan, A. Bianco (Eds.), Quality of Service in Multiservice IP Networks. Proceedings, 2001. XII, 440 pages. 2001.

Vol. 1990: I.V. Ramakrishnan (Ed.), Practical Aspects of Declarative Languages. Proceedings, 2001. VIII, 353 pages. 2001.

Vol. 1991: F. Dignum, C. Sierra (Eds.), Agent Mediated Electronic Commerce. VIII, 241 pages. 2001. (Subseries LNAI).

Vol. 1992: K. Kim (Ed.), Public Key Cryptography. Proceedings, 2001. XI, 423 pages. 2001.

Vol. 1993: E. Zitzler, K. Deb, L. Thiele, C.A.Coello Coello, D. Corne (Eds.), Evolutionary Multi-Criterion Optimization. Proceedings, 2001. XIII, 712 pages. 2001.

Vol. 1995: M. Sloman, J. Lobo, E.C. Lupu (Eds.), Policies for Distributed Systems and Networks. Proceedings, 2001. X, 263 pages. 2001.

Vol. 1997: D. Suciu, G. Vossen (Eds.), The World Wide Web and Databases. Proceedings, 2000. XII, 275 pages. 2001.

Vol. 1998: R. Klette, S. Peleg, G. Sommer (Eds.), Robot Vision. Proceedings, 2001. IX, 285 pages. 2001.

Vol. 1999: W. Emmerich, S. Tai (Eds.), Engineering Distributed Objects. Proceedings, 2000. VIII, 271 pages. 2001.

Vol. 2000: R. Wilhelm (Ed.), Informatics: 10 Years Back, 10 Years Ahead. IX, 369 pages. 2001.

Vol. 2003: F. Dignum, U. Cortés (Eds.), Agent Mediated Electronic Commerce III. XII, 193 pages. 2001. (Subseries LNAI).

Vol. 2004: A. Gelbukh (Ed.), Computational Linguistics and Intelligent Text Processing. Proceedings, 2001. XII, 528 pages. 2001.

Vol. 2006: R. Dunke, A. Abran (Eds.), New Approaches in Software Measurement. Proceedings, 2000. VIII, 245 pages. 2001.

Vol. 2007: J.F. Roddick, K. Hornsby (Eds.), Temporal, Spatial, and Spatio-Temporal Data Mining. Proceedings, 2000. VII, 165 pages. 2001. (Subseries LNAI).

Vol. 2009: H. Federrath (Ed.), Designing Privacy Enhancing Technologies. Proceedings, 2000. X, 231 pages. 2001.

Vol. 2010: A. Ferreira, H. Reichel (Eds.), STACS 2001. Proceedings, 2001. XV, 576 pages. 2001.

Vol. 2013: S. Singh, N. Murshed, W. Kropatsch (Eds.), Advances in Pattern Recognition – ICAPR 2001. Proceedings, 2001. XIV, 476 pages. 2001.

Vol. 2015: D. Won (Ed.), Information Security and Cryptology – ICISC 2000. Proceedings, 2000. X, 261 pages. 2001.

Vol. 2018: M. Pollefeys, L. Van Gool, A. Zisserman, A. Fitzgibbon (Eds.), 3D Structure from Images – SMILE 2000. Proceedings, 2000. X, 243 pages. 2001.

Vol. 2020: D. Naccache (Ed.), Progress in Cryptology – CT-RSA 2001. Proceedings, 2001. XII, 473 pages. 2001

Vol. 2021: J. N. Oliveira, P. Zave (Eds.), FME 2001: Formal Methods for Increasing Software Productivity. Proceedings, 2001. XIII, 629 pages. 2001.

Vol. 2024: H. Kuchen, K. Ueda (Eds.), Functional and Logic Programming. Proceedings, 2001. X, 391 pages. 2001.

Vol. 2027: R. Wilhelm (Ed.), Compiler Construction. Proceedings, 2001. XI, 371 pages. 2001.

Vol. 2028: D. Sands (Ed.), Programming Languages and Systems. Proceedings, 2001. XIII, 433 pages. 2001.

Vol. 2029: H. Hussmann (Ed.), Fundamental Approaches to Software Engineering. Proceedings, 2001. XIII, 349 pages. 2001.

Vol. 2030: F. Honsell, M. Miculan (Eds.), Foundations of Software Science and Computation Structures. Proceedings, 2001. XII, 413 pages. 2001.

Vol. 2031: T. Margaria, W. Yi (Eds.), Tools and Algorithms for the Construction and Analysis of Systems. Proceedings, 2001. XIV, 588 pages. 2001.

Vol. 2034: M.D. Di Benedetto, A. Sangiovanni-Vincentelli (Eds.), Hybrid Systems: Computation and Control. Proceedings, 2001. XIV, 516 pages. 2001.

Vol. 2035: D. Cheung, G.J. Williams, Q. Li (Eds.), Knowledge Discovery and Data Mining – PAKDD 2001. Proceedings, 2001. XVIII, 596 pages. 2001. (Subseries LNAI).

Vol. 2038: J. Miller, M. Tomassini, P.L. Lanzi, C. Ryan, A.G.B. Tettamanzi, W.B. Langdon (Eds.), Genetic Programming. Proceedings, 2001. XI, 384 pages. 2001.

Table of Contents

Talks

Posters

Heuristic Learning Based on Genetic Programming

Nicole Drechsler, Frank Schmiedle, Daniel Große, and Rolf Drechsler

Institute of Computer Science
Chair of Computer Architecture (Prof. Bernd Becker)
Albert-Ludwigs-University
79110 Freiburg im Breisgau
ndrechsl@informatik.uni-freiburg.de

Abstract. In this paper we present an approach to learning heuristics based on Genetic Programming (GP). Instead of directly solving the problem by application of GP, GP is used to develop a heuristic that is applied to the problem instance. By this, the typical large runtimes of evolutionary methods have to be invested only once in the learning phase. The resulting heuristic is very fast. The technique is applied to a field from the area of VLSI CAD, i.e. minimization of Binary Decision Diagrams (BDDs). We chose this topic due to its high practical relevance and since it matches the criteria where our algorithm works best, i.e. large problem instances where standard evolutionary techniques cannot be applied due to their large runtimes. Our experiments show that we obtain high quality results that outperform previous methods, while keeping the advantage of low runtimes.

1 Introduction

Decision Diagrams (DDs) are often used in CAD systems for efficient representation and manipulation of Boolean functions. The most popular data structure is the *Binary Decision Diagram* (BDD) [Bry86]. Recently, several approaches in logic synthesis have been presented that make use of BDDs. (For an overview see [DB98].) One drawback of this data structure is that it is very sensitive to the variable ordering, i.e. the size may vary from linear to exponential. Finding the optimal variable ordering is a difficult problem [BW96] and the best known algorithm has exponential runtime.

This is the reason why in the last few years many authors presented heuristics for finding good variable orderings. The most promising methods are based on dynamic variable ordering [FMK91],[Rud93],[PS95]: BDDs for some Boolean functions have been constructed for which all other topology oriented methods failed. New methods based on non-deterministic algorithms have been proposed for BDD minimization, e.g. genetic algorithms [DBG95] and simulated annealing [RBKM92],[BLW95]. The major drawback of these approaches is that in general they obtain good results with respect to quality of the solution, but the running times are often much larger than those of classical heuristics. Due to the high complexity of the design process in VLSI CAD often fast heuristics are used. These heuristics are developed by the designer

J. Miller et al. (Eds.) : EuroGP 2001, LNCS 2038, pp. 1-10, 2001.

himself. But they also often fail for specific classes of circuits. Thus it would help a lot, if the heuristics could learn from previous examples, e.g. from benchmark examples.

A theoretical model for learning heuristics by *Genetic Algorithms* (GAs) has been presented in [DB95]. The new aspect of this model is that the GA is not directly applied to the problem. Instead the GA develops a good heuristic for the problem to be solved. First applications to multi-level synthesis and to 2-level AND/EXOR minimization have been presented. There the model has not been fully used, i.e. only a part of the features has been implemented. Extensions have been proposed in [DGB96] and [DDB99], where learning of BDD heuristics has been studied based on GAs. But due to the fixed length encoding these approaches also have some disadvantages:

- The length of the heuristic is limited resulting in limitations with respect to quality.
- Decision procedures, like if-then-else, could not be integrated in the heuristics.

In this paper we present an approach to heuristic learning based on *Genetic Programming* (GP) [Koz92],[Koz94]. Due to the more flexible encoding based on tree structures, the disadvantages of the GA approaches described above can be avoided, while keeping the advantages. To keep the heuristic under development as compact as possible, reduction operators are introduced. Compared to GAs more flexible operators are defined and integrated in the GP run. Experimental results demonstrate the efficiency of the approach.

The paper is structured as follows: In the next section the problem definition is given and the model of heuristic learning is described. Then the solution based on GP is outlined and the (genetic) operators are introduced. Experimental results are reported and finally the paper is summarized.

2 Problem Description

In [DB95] a learning model has formally been introduced for GAs. In this section we briefly review the main notation and definitions to make the paper self-contained. Since the definition of the model is based on the evaluation of the fitness function only, it becomes obvious that even though it has been developed for GAs, it can be transferred to GPs directly.

It is assumed that the problem to be solved has the following property: There is defined a non empty set of optimization procedures that can be applied to a given (non-optimal) solution in order to further improve its quality. These procedures are called *Basic Optimization Modules* (BOMs) in the following. The heuristics are sequences of BOMs. The goal of the approach is to determine a good (or even optimal) sequence of BOMs such that the overall results obtained by the heuristics are improved. From the flexibility of the tree-like data structure in GPs we are later even able to define more powerful operators.

In the following we assume that the reader is familiar with the basic concepts and notation of evolutionary approaches. In [DB95] and [DGB96], a multi-valued string encoding of fixed length has been used, but this already by definition limits the quality of the result. Due to the GP concept, there is in principle no limit on the size of the resulting heuristics.

The set of BOMs defines the set H of all possible heuristics that are applicable to the problem to be solved in the given environment. H may include problem specific heuristics but can also include randomized techniques.

To each BOM h we associate a cost function, i.e. a value that determines how expensive it is to call this module. This value can be chosen dependent on different criteria, like memory consumption or run time. Furthermore, the quality function determines the quality resulting from the application of this module. These two values determine the fitness of the BOM. Summation over all BOMs in a heuristic determines the overall fitness. The elements are evaluated on a set of benchmark examples, the so-called training set. The multiple objectives are optimized in parallel by the method described in [DDB99].

2.1 BOMs

Most of the algorithms that are used here as BOMs for heuristic learning are well-known BDD minimization techniques They are based on dynamic variable ordering. *Sifting* (SIFT) is a local search operation for variable ordering of BDDs which allows hill climbing. *Group sifting* (GROUP) and *symmetric sifting* (SYMM) additionally make use of symmetry aspects of the variables of the considered Boolean functions. The BOM *window permutation* of size 3 (4), denoted as WIN3 (WIN4), tests all permutations of 3 (4) adjacent variables, where the window of size 3 (4) "goes" through the whole BDD from the top to the bottom variables. For all these techniques there is an additional BOM that iterates the method until convergence is reached. These BOMs are denoted by the appendix *CO, e.g.* iterated sifting is denoted by SIFTCO. The next BOM is called *inversion* (I) and inverts the variable ordering of a BDD between two randomly chosen variables. Finally, the set of BOMs contains an "empty" element, denoted as NOOP. In the GA approach this operator was used to model strings of various sizes, while the underlying data structure was a fixed-length string. Using GPs this problem by definition does not occur. Nevertheless, NOOP is important to describe a condition, e.g. it is possible to describe an IF-condition without a statement in the ELSE-branch. These types of operators could not be introduced using Gas, underlining the flexibility of the approach introduced below.

For more details about the learning model, the definition of BDD, and BOMs see [DB95], [DGB96], and [DDB99].