

Lecture Notes in Artificial Intelligence

886

Subseries of Lecture Notes in Computer Science

Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis and J. van Leeuwen



Manuela M. Veloso

Planning and Learning by Analogical Reasoning

Springer-Verlag

Berlin Heidelberg New York

London Paris Tokyo

Hong Kong Barcelona

Budapest

Series Editors

Jaime G. Carbonell
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3891, USA

Jörg Siekmann
University of Saarland
German Research Center for Artificial Intelligence (DFKI)
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany

Author

Manuela M. Veloso
Department of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3891, USA
veloso@cs.cmu.edu

CR Subject Classification (1991): I.2.6, I.2.8, I.2.3

ISBN 3-540-58811-6 Springer-Verlag Berlin Heidelberg New York

CIP data applied for

This work is subject to copyright. All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, 1965, in its current version, and permission for use must always be obtained from Springer-Verlag. Violations are liable for prosecution under the German Copyright Law.

© Springer-Verlag Berlin Heidelberg 1994
Printed in Germany

Typesetting: Camera ready by author
SPIN: 10479112 45/3140-543210 - Printed on acid-free paper

Preface

This book describes the integration of analogical reasoning into general problem solving as a method of learning at the strategy level to solve problems more effectively. The method, based on derivational analogy, has been fully implemented in *PRODIGY/ANALOGY* and proven empirically to be amenable to scaling up both in terms of domain and problem complexity.

Reasoning by analogy involves a set of challenging problems, namely: how to accumulate episodic problem solving experience, how to define and decide when two problem solving situations are similar, how to organize large amounts of episodic knowledge so that it may be efficiently retrieved, and finally the ultimate problem of how to successfully transfer chains of reasoning from past experience to new problem solving situations when only a partial match exists among the corresponding problems.

In this work, the strategy-level learning process is cast for the first time as the automation of the complete cycle of constructing, storing, retrieving, and flexibly reusing problem solving experience. This book presents the interconnected algorithms designed for the generation, storage, retrieval and replay of derivational traces of problem solving episodes. The integrated learning system reduces the problem solving search effort incrementally as more episodic experience is compiled into the library of accumulated learned knowledge.

A primary goal of this work was to demonstrate an integrated view of the flexible learning and problem solving methods in large and complex domains. Scaling up the system proved to be very demanding. The current system has thus far been demonstrated in multiple domains, including a complex logistics transportation domain where it generated a library of 1000 cases. In addition to the analogical reasoning method, the book presents empirical results showing how *PRODIGY/ANALOGY* improved the problem-solving performance many-fold, actually increased the quality of the solutions generated, and pushed the solvability envelope to increasingly more complex classes of nonlinear planning problems.

Acknowledgements

This book is based on my PhD thesis done at Carnegie Mellon University (CMU). Special thanks are due to my adviser, Jaime Carbonell, with whom this work was done in close collaboration. Jaime shared with me many of his ideas and helped me solve many problems I faced along this work. He gave me extremely insightful advice.

I would like to thank my other thesis committee members, Tom Mitchell, Paul Rosenbloom, and Herb Simon, for all the helpful suggestions and comments on my research. Herb Simon and Tom Mitchell followed my work very closely. Herb Simon was always interested on the detailed progress of my work. He encouraged me to reach a full implementation of the system, so I could scale it up to tasks of realistic size. Thanks to Herb Simon, I have a much broader view of this research. With him I learned to appreciate many of the early research efforts in Artificial Intelligence especially in learning. In our several meetings, Tom Mitchell helped me keeping focused on my thesis goals, at the same time that he opened my interests to other related issues in machine learning. He raised challenging questions while listening carefully to my ideas and discussing his own with me. Through several email conversations, Paul Rosenbloom provided me very useful feedback on my work. He helped me clarify some of my claims of this work.

Allen Newell was not in my thesis committee. Still he shaped my thinking. We had several long meetings where we discussed my work. Allen Newell was always there pointing out the interesting points, forcing me to clarify my ideas, making sure I myself would understand well the concepts, the algorithms, the implementation. From Allen Newell I learned every day more and more what research is. Through him I understood how to work dynamically towards my research goals. He was a real mentor and friend.

I gratefully acknowledge also all the members of the PRODIGY project, in particular Craig Knoblock, Alicia Pérez, Daniel Borrajo, Steve Minton, Yolanda Gil, Oren Etzioni, Robert Joseph, Jim Blythe, Xuemei Wang, and Eugene Fink, for their helpful comments and suggestions on the work and on the dissertation. I thank all my friends at CMU for their support. In a very special way, I thank Puneet Kumar. I benefited tremendously from his ceaseless help with system changes and enhancements.

The School of Computer Science at CMU is a very special place. I acknowledge Nico Habermann for having consistently promoted a community spirit

among the members of this large research group. I thank Sharon Burks for her constant willingness to make more pleasant and facilitate our students' lives.

Last, but most importantly, I would like to thank all my family for their loving support. Back in Lisbon, my parents, my parents-in-law, and all my family courageously supported and encouraged this work, in spite of the inevitable long separation. I thank them all for this. I am grateful to my parents and parents-in-law for making our returns home on vacation so wonderful, full of the little things they know we miss so much. Thank you for all their letters, their telephone calls, their visits, thank you for their love that really carried me through these years. I am very thankful also for their help taking care of my sons, André and Pedro, who love so much going to Portugal,

I thank my husband Manel, and my sons, André and Pedro, for having lived with me through the thesis work. They were always here to love me, encourage me, and make me feel that it was all worth the effort. In particular I want to thank Pedro and André for the many weekends that we spent together at CMU. They were always cheerful and packed their books and toys happily for the CMU journeys without any complaint. Also I would like to thank André and Pedro for the after-school afternoons that I could not spend with them. I waited eagerly for André's daily phone call when he got home from school, ready and responsible to stay home alone during the afternoon. It would have been very hard to complete this thesis without Manel's support. I thank him also in particular for his surprise on the day of my defense.

The work reported in this book was supported by the Avionics Laboratory, Wright Research and Development Center, Aeronautical Systems Division (AFSC), U.S. Air Force, Wright-Patterson AFB, Ohio 45433-6543 under Contract F33615-90-C-1465, ARPA Order No. 7597. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or of the U.S. Government.

Contents

1	Introduction	1
1.1	Machine learning and problem solving	2
1.2	Analogy within PRODIGY	5
1.3	The thesis of this book	8
1.4	Reader's guide	11
2	Overview	15
2.1	Case generation	15
2.1.1	Defining a problem	15
2.1.2	Problem solving search	16
2.1.3	Justification structure	19
2.2	Case storage	20
2.2.1	Foot-printing the initial state	21
2.2.2	Multiple-goal problems	23
2.3	Automatic case retrieval	27
2.4	Case replay	28
2.5	Summary	30
3	The Problem Solver	33
3.1	Motivation	33
3.1.1	Linear problem solving	34
3.1.2	Nonlinear problem solving	37
3.2	NOLIMIT - The planning algorithm	39
3.3	Formal problem solving procedure	41
3.3.1	Failing and backtracking	44
3.3.2	Control knowledge	46
3.4	An example: a <i>one-way-rocket</i> problem	47
3.5	Summary	51
4	Generation of Problem Solving Cases	53
4.1	Annotating the search path	53
4.1.1	The decision points at problem solving	54
4.1.2	Justification structures at decision nodes	56
4.1.3	The language	57
4.2	The annotation procedure	58
4.2.1	Annotating the subgoaling structure	59
4.2.2	Annotating the failures	60
4.3	An example in the extended-STRIPS domain	61
4.4	Summary	65

5	Case Storage: Automated Indexing	67
5.1	Identifying independent case subparts	68
5.1.1	Transforming a total order into a partial order	69
5.1.2	Goal indices	71
5.2	Identifying the relevant initial state	74
5.2.1	Disambiguating the notion of "relevant"	74
5.2.2	Foot-printing the initial state	77
5.3	Organization of the case library	79
5.4	The complete storage algorithm	86
5.5	Summary	90
6	Efficient Case Retrieval	91
6.1	The ground for the retrieval procedure	91
6.1.1	What are similar problem solving situations?	93
6.1.2	How can retrieval be efficient in a large case library?	93
6.2	Defining a similarity metric	95
6.2.1	A direct similarity metric	96
6.2.2	Global foot-printing similarity metric	96
6.2.3	Interacting foot-printing similarity metric	97
6.3	The retrieval procedure	98
6.3.1	Indexing data structures	100
6.3.2	Illustrative example	101
6.4	Trading off retrieval and search costs	107
6.5	Summary	110
7	Analogical Replay	111
7.1	Replaying past problem solving episodes	111
7.1.1	Outline of the replay procedure	112
7.1.2	Advantages of replaying	113
7.1.3	Feedback to memory	116
7.2	The replay algorithm	117
7.2.1	Generation of new search directions	118
7.2.2	Pursuing the search	123
7.2.3	Advancing the cases	129
7.3	Examples	131
7.4	Feedback: problem solver to memory	136
7.4.1	The method explored	136
7.4.2	Illustrative example	137
7.5	Summary	138

8 Empirical Results	141
8.1 Diversity of tasks	142
8.1.1 The <i>one-way-rocket</i> domain	143
8.1.2 The extended-STRIPS and machine-shop domains	144
8.2 Scale up: A logistics transportation domain	146
8.2.1 Generation of problems	146
8.2.2 Set up of experiments	147
8.2.3 The solvability horizon	149
8.2.4 Cumulative running times	153
8.2.5 Solution length	154
8.2.6 Retrieval and replay times	156
8.2.7 Retrieval time against the size of the case library	159
8.2.8 Search nodes explored	160
8.3 Summary	161
9 Related Work	163
9.1 Generation and contents of cases	163
9.2 Storage and retrieval of cases	164
9.3 Utilization of learned knowledge	166
9.4 Summary	168
10 Conclusion	169
11 Bibliography	173