

Lecture Notes in Artificial Intelligence 2854

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

Springer

Berlin

Heidelberg

New York

Hong Kong

London

Milan

Paris

Tokyo

Jörg Hoffmann

Utilizing Problem Structure in Planning

A Local Search Approach



Springer

Series Editors

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

Author

Jörg Hoffmann
Universität Freiburg, Institut für Informatik
Georges-Köhler-Allee, Geb. 52, 79110 Freiburg, Germany
E-mail: hoffmann@informatik.uni-freiburg.de

Cataloging-in-Publication Data applied for

A catalog record for this book is available from the Library of Congress

Bibliographic information published by Die Deutsche Bibliothek

Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliographie;
detailed bibliographic data is available in the Internet at <<http://dnd.ddb.de>>.

CR Subject Classification (1998): I.2.8, I.2, F.2.2

ISSN 0302-9743

ISBN 3-540-20259-5 Springer-Verlag Berlin Heidelberg New York

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 New York,
a member of BertelsmannSpringer Science+Business Media GmbH

<http://www.springer.de>

© Springer-Verlag Berlin Heidelberg 2003
Printed in Germany

Typesetting: Camera-ready by author, data conversion by Christian Grosche, Hamburg
Printed on acid-free paper SPIN: 10965696 06/3142 5 4 3 2 1 0

Foreword

Planning is a crucial skill for any autonomous agent, be it a physically embedded agent, such as a robot, or a purely simulated, software agent. For this reason, planning is one of the central research areas in Artificial Intelligence. The area is concerned with the automatic construction of plans that map a given initial situation into a target situation that satisfies some given goal conditions. Here, plans are usually sequences of deterministic actions.

While there has been a lot of research in this area since the beginning of the 1970s, only recently has the practical efficiency of planning methods come into the focus of the research community – most probably partially caused by the biennial international competition of planning systems, which started in 1998.

Planning is known to be a computationally quite demanding problem. Even in its simplest form – planning for basic propositional STRIPS – planning is PSPACE-complete. However, the planning problems humans and artificial agents are asked to solve are often much easier than the general problem. Based on this observation, Jörg Hoffmann investigated the structure of such planning tasks. Starting with the planning tasks used in the first planning competition, he developed his planning methodology to be able to cope with these and other problems published in the planning literature. So, instead of developing a new general planning method that might also solve the problems at hand, Jörg developed his planning methodology having in mind the concrete planning tasks.

The main ingredient of Jörg's planning methodology is the view of planning as a heuristic search, as developed by Geffner and Bonet. This approach is based on extracting informative heuristics, i.e., estimations of a search state's goal distance, automatically from the problem encoding.

Based on this work, Jörg develops a new heuristic that takes into account positive interactions between different actions, he designs a new local search technique, and he introduces a new pruning technique. These three techniques are the main methods used in the planning system FF, whose performance, measured on a large range of benchmark problems, is well above most of the other known planning systems. In fact, FF was the clear winner of the international planning competition in 2000 and is now one of the main references in the area of planning.

This success raised the question of what the structural properties of the benchmark problems are that allow us to solve them so easily. Based on earlier work in the area of propositional satisfiability, Jörg analyzes the landscape of the problems that are induced by the heuristic estimate. He is able to show

that the empirical hardness is dependent on the number and size of local minima and plateaus and on the number and distance of improved exits for the latter. The analysis is carried out both theoretically and empirically, and divides the current planning benchmarks into a taxonomy of classes that differ in terms of how easily they can be solved using heuristic planners such as FF. These results are very useful for evaluating domains and for improving local search algorithms.

In summary, this work's two contributions to the state of the art in planning are so significant that the dissertation this book is based on received the prestigious ECCAI award for the best European Ph.D. thesis in 2002.

July 2003

Bernhard Nebel

Acknowledgements

This book is a revised version of my doctoral thesis, submitted to the Albert-Ludwigs-Universität Freiburg. It would not have been written without the support, encouragement, and advice of a large number of people, whom I would like to thank.

I am indebted to my thesis advisor Bernhard Nebel, whose advice was crucial for taking all the research decisions necessary to obtain the results contained in this book. I am also thankful to the second reader of my thesis, Héctor Geffner, whose work on the HSP system inspired my work on FF in the first place. I thank Jana Koehler, from whom I learned loads of things while I was a student member of the IPP team under her lead.

I would like to thank the people in the AI planning community at large; meeting them was always interesting, and fun. Special thanks go to Maria Fox, Derek Long, Julie Porteous, Laura Sebastia, Eva Onaindia, Antonio Garrido, Carmel Domshlak, Ronen Brafman, Blai Bonet, Sylvie Thiebaux, Bart Selman, Carla Gomes, Fahiem Bacchus, and many others, for enjoyable visits, fun nights out, inspiring discussions, and/or fruitful collaborative work. A very special mention goes to Eva Onaindia and Carmel Domshlak, who let me stay in the guestrooms of their private apartments during visits to Valencia University in March 2001 and to Cornell University in May 2003, respectively.

I would like to thank my colleagues in the AI research group in Freiburg, for discussions, visiting conferences together, and generally for making working life enjoyable. Special thanks go to Malte Helmert for numerous useful discussions, and to Jussi Rintanen for reading an early version of my thesis. Very special thanks go to my office mate Thilo Weigel, for many humorous comments, for watering the plants, for suggesting to put a new line into the title of this book, and for singing along to Linkin Park and Britney Spears. Sitting in the office would have been boring without him.

I dedicate this book to my family and friends.

Table of Contents

Part I. Planning: Motivation, Definitions, Methodology

1. Introduction	3
1.1 Planning, Problem Structure, and Local Search	3
1.2 An Illustrative Example	5
1.3 Outline	8
2. Planning	11
2.1 Motivation, Goals, and History	11
2.2 STRIPS and ADL	13
2.2.1 Predicates, Facts, and Groundings	14
2.2.2 States, Actions, and Tasks	14
2.2.3 Operators, Instances, and Domains	14
2.2.4 Plans	15
2.2.5 Extension to ADL	17
2.3 The Benchmark Domains	19
2.3.1 Transportation Domains	20
2.3.2 Construction Domains	22
2.3.3 Other Domains	24
2.4 Previous Approaches	25
2.4.1 Partial Order Planning	25
2.4.2 Planning Graph Analysis	27
2.4.3 Planning as Satisfiability	29

Part II. A Local Search Approach

3. Base Architecture	35
3.1 Grounding	41
3.1.1 STRIPS	41
3.1.2 ADL	42
3.2 HSP	47
3.2.1 Heuristic	47
3.2.2 Search	48
3.2.3 Evaluation	49

3.3	Heuristic	51
3.3.1	Planning Graphs for Relaxed Tasks	51
3.3.2	Solution Length Optimization	54
3.3.3	Efficient Implementation	57
3.4	Search	58
3.4.1	Enforced Hill-Climbing	58
3.4.2	Completeness	59
3.5	Pruning	60
3.5.1	Helpful Actions	60
3.5.2	Completeness	62
3.6	Extension to ADL	63
3.6.1	Heuristic Function	63
3.6.2	Pruning	64
3.7	Evaluation	64
3.8	What Makes the Difference to HSP?	66
3.8.1	Experimental Setup	67
3.8.2	Runtime	67
3.8.3	Solution Length	71
4.	Dead Ends	75
4.1	Dead-End Free Tasks	77
4.1.1	Undirected State Spaces	79
4.1.2	Harmless State Spaces	80
4.1.3	Practical Implications	82
4.2	Handling Dead Ends	84
4.2.1	Restarts	84
4.2.2	A Safety Net Solution	86
4.3	Evaluation	87
5.	Goal Orderings	89
5.1	Pre-Computing Goal Orderings	90
5.1.1	Recognizing Goal Orderings	91
5.1.2	An Agenda-Driven Planning Algorithm	94
5.1.3	Completeness, and Integration into FF	96
5.2	Computing Goal Orderings On-Line	97
5.2.1	Untimely Goals	97
5.2.2	Approximating Untimely Goals	98
5.2.3	Completeness, and Integration into FF	99
5.3	Evaluation	100
6.	The AIPS-2000 Competition	107
6.1	The <i>Logistics</i> Domain	107
6.2	The <i>Blocksworld-arm</i> Domain	108
6.3	The <i>Schedule</i> Domain	109
6.4	The <i>Freecell</i> Domain	110
6.5	The <i>Miconic</i> Domain	110

Part III. Local Search Topology

7. Gathering Insights	115
7.1 Methodology	120
7.2 The Topology of h^+	121
7.2.1 Dead Ends	122
7.2.2 Local Minima	124
7.2.3 Benches	125
7.2.4 A Planning Domain Taxonomy	128
7.3 The Topology of h^{FF}	131
7.3.1 Local Minima	131
7.3.2 Benches	131
7.3.3 The Hypotheses	132
7.4 Visualization	133
8. Verifying the h^+ Hypotheses	135
8.1 A Structural Core	136
8.2 No Local Minima and Bounded Maximal Exit Distance	140
8.2.1 Simple-Tsp	140
8.2.2 Movie	141
8.2.3 Gripper	142
8.2.4 Ferry	143
8.2.5 Logistics	145
8.2.6 Miconic-STRIPS	146
8.2.7 Miconic-SIMPLE	147
8.3 No Local Minima	149
8.3.1 Tireworld	149
8.3.2 Briefcaseworld	152
8.3.3 Fridge	154
8.3.4 Blocksworld-no-arm	156
8.3.5 Grid	158
8.3.6 Hanoi	161
8.4 No Unrecognized Dead Ends	163
8.4.1 Blocksworld-arm	163
8.4.2 Schedule	164
8.5 Assembly	165
8.5.1 Solvability	168
8.5.2 Some Observations Concerning Dead Ends	170
8.5.3 Some Observations Concerning Local Minima	175
8.6 The Taxonomy	179

9. Supporting the h^{FF} Hypotheses	181
9.1 Methodology	181
9.2 Domains with a Single Parameter	185
9.3 Domains with Two Parameters	187
9.3.1 Briefcaseworld	187
9.3.2 Ferry	188
9.3.3 Fridge	189
9.3.4 Miconic-SIMPLE	189
9.3.5 Miconic-STRIPS	190
9.4 Domains with Multiple Parameters	191
9.4.1 Logistics	191
9.4.2 Grid	193
9.5 Results Overview	196
10. Discussion	199
A. Formalized Benchmark Domains	203
A.1 Assembly	203
A.2 Blocksworld-arm	206
A.3 Blocksworld-no-arm	207
A.4 Briefcaseworld	208
A.5 Ferry	209
A.6 Freecell	210
A.7 Fridge	213
A.8 Grid	214
A.9 Gripper	216
A.10 Hanoi	217
A.11 Logistics	217
A.12 Miconic-ADL	219
A.13 Miconic-SIMPLE	221
A.14 Miconic-STRIPS	222
A.15 Movie	223
A.16 Mprime	224
A.17 Mystery	226
A.18 Schedule	226
A.19 Simple-Tsp	231
A.20 Tireworld	231
B. Automated Instance Generation	235
B.1 Assembly	236
B.2 Blocksworld-arm	237
B.3 Blocksworld-no-arm	237
B.4 Briefcaseworld	237
B.5 Ferry	237
B.6 Freecell	237
B.7 Fridge	238

B.8 Grid	238
B.9 Gripper	238
B.10 Hanoi	238
B.11 Logistics	239
B.12 Miconic-ADL	239
B.13 Miconic-SIMPLE	240
B.14 Miconic-STRIPS	240
B.15 Movie	240
B.16 Mprime	240
B.17 Mystery	240
B.18 Schedule	241
B.19 Simple-Tsp	241
B.20 Tireworld	241

List of Figures

1.1	Informal example of a planning task	6
2.1	Runtime (t) and solution length (s) results for UCPOP in our benchmark example collection. A dash means that no plan was found after 5 minutes. An empty entry means that the planner could not be run on the respective task. Some domain names are abbreviated in order to fit the figure into the page without making it too tiny to read	26
2.2	Runtime (t) and solution length (s) results for IPP in our benchmark example collection. A dash means that no plan was found after 5 minutes	29
2.3	Runtime (t) and solution length (s) results for Blackbox in our benchmark example collection. A dash means that no plan was found after 5 minutes. An empty entry means that the planner could not be run on the respective task	30
3.1	Runtime (t) and solution length (s) results of our HSP1 implementation, averaged over five trials. A dash means that no trial found a plan after 5 minutes	50
3.2	Relaxed plan extraction for a task (A^+, s, G)	58
3.3	The enforced hill-climbing algorithm, for a task with heuristic h	59
3.4	Runtime (t) and solution length (s) results of the FF base architecture. A dash means that no plan was found after 5 minutes	65
3.5	Averaged runtime per domain for all eight configurations of switches .	68
3.6	The effect of turning on a single switch, keeping the others unchanged. Summarized in terms of significantly improved or degraded running time performance per domain, and per switch configuration	70
3.7	The effect of turning on a single switch, keeping the others unchanged. Summarized in terms of significantly improved or degraded solution length performance per domain, and per switch configuration	72
4.1	Runtimes for different configurations of FF on the AIPS-1998 <i>Mystery</i> collection (instances that can be proved unsolvable by IPP are left out of the table). A dash means that no plan was found after 5 minutes . .	85

4.2	Runtime for the FF base architecture, respectively that architecture including safety net, on example collections from domains where the FF base architecture sometimes fails. Times in parentheses specify the time after which failure was reported. A dash means that no plan was found (no failure was reported) after 5 minutes	87
5.1	Runtime curves on <i>Blocksworld-arm</i> for FF with different goal ordering techniques. Time is shown on a logarithmic scale. If a task could not be solved within the given time and memory then the data point is omitted	101
5.2	Runtime curves on <i>Blocksworld-no-arm</i> for FF with different goal ordering techniques. Time is shown on a logarithmic scale. If a task could not be solved within the given time and memory then the data point is omitted	102
5.3	Runtime curves on <i>Miconic-ADL</i> for FF with different goal ordering techniques. Time is shown on a logarithmic scale. If a task could not be solved within the given time and memory then the data point is omitted	103
5.4	Runtime curves on <i>Schedule</i> for FF with different goal ordering techniques. Time is shown on a logarithmic scale. If a task could not be solved within the given time and memory then the data point is omitted	104
6.1	Runtime curves on large <i>Logistics</i> instances for those six planners that could scale up to them. Time is shown on a logarithmic scale	108
6.2	Runtime curves on large <i>Blocksworld-arm</i> instances for those three planners that could scale up to them: FF, HSP2, and System-R. Time is shown on a logarithmic scale	109
6.3	Runtime curves on <i>Schedule</i> instances for those planners that could handle conditional effects. Time is shown on a logarithmic scale.	110
6.4	Runtime curves on <i>Freecell</i> tasks for those planners that scaled to larger instances. Time is shown on a logarithmic scale	111
6.5	Runtime curves for those planners that participated in <i>Miconic-ADL</i> . Time is shown on a logarithmic scale.	111
7.1	Percentage of states with h^+ value less than infinity in the state space, i.e., relevant part percentage. Mean values for increasing number of constants in different domains	122
7.2	Percentage of unrecognized dead ends in the relevant part of the state space under h^+ . Mean values for increasing number of constants in different domains	123
7.3	Percentage of states on valleys in the relevant part of the state space under h^+ . Mean values for increasing number of constants in different domains	124

7.4	Maximal size (upper part) and diameter (lower part) of valley regions under h^+ . Mean values for increasing number of constants in different domains	125
7.5	Percentage, under h^+ , of states on bench-related plateaus that are not part of a valley. Mean values for increasing number of constants in different domains	126
7.6	Maximal exit distance under h^+ . Mean values for increasing number of constants in different domains	127
7.7	The planning domain taxonomy, overviewing our hypotheses	128
7.8	Percentage of states on valleys in the relevant part of the state space under h^{FF} . Mean values for increasing number of constants in different domains	131
7.9	Maximal exit distance under h^{FF} . Mean values for increasing number of constants in different domains	132
7.10	Visualized state space under h^+ of (a) the <i>Gripper</i> instance with 4 balls and (b) a <i>Logistics</i> instance with 2 cities (each 3 locations and 1 truck), 1 object, and 1 airplane	133
7.11	Visualized state space under (a) h^+ and (b) h^{FF} of the <i>Hanoi</i> instance with 6 discs	134
8.1	An <i>Assembly</i> instance where an unrecognized dead end arises in the situation where G is incorporated into D , and I is incorporated into C	171
8.2	An <i>Assembly</i> instance where a local minimum arises when D is incorporated into B , and G is incorporated into C	175
8.3	The proved planning domain taxonomy, overviewing our results	179
9.1	Percentage of sample states on valleys. Mean values for increasing number of constants in different domains, in the example collection from Chapter 7. Real values are in parentheses	184
9.2	Percentage of sample states on valleys. Mean values for increasing number of constants in different domains	186
9.3	Maximal exit distance. Mean values for increasing number of constants in different domains	186
9.4	Valley percentage (z -axis) in <i>Briefcaseworld</i> instances, scaling locations (x -axis) against portables (y -axis)	188
9.5	Maximal exit distance (z -axis) in <i>Fridge</i> instances, scaling fridges (x -axis) against screws (y -axis)	189
9.6	Sampled average valley percentage (a) and maximal exit distance (b) in <i>Miconic-SIMPLE</i> . Data is shown on the z -axis, scaling floors (x -axis) against passengers (y -axis)	190

9.7	Sampled average valley percentage (left hand side) and maximal exit distance (right hand side) in <i>Logistics</i> , when scaling cities against objects (a and b), cities against airplanes (c and d), and objects against airplanes (e and f). Data is shown on the z -axis; when scaling domain parameter A against domain parameter B, the value of A is depicted on the x -axis, and the value of B is depicted on the y -axis	192
9.8	Valley percentage in <i>Grid</i> instances, scaling x -extension (x -axis) against keys (y -axis). There are two different key and lock shapes, y -extension is fixed to 3, number of locks is fixed to 3/2	194
9.9	Valley percentage in <i>Grid</i> instances, scaling x -extension (x -axis) against locks (y -axis). There are two different key and lock shapes, y -extension is fixed to 3, number of keys is fixed to 3/2	195
9.10	How the h^+ properties carry over to h^{FF} , summarized in terms of the taxonomy paradigm	196