

Catalog of Planning Resources I: An Annotated Bibliography of Classical Planning

Contents

I INTRODUCTION	8
1 Acknowledgments	8
2 Copyright and Colophon	9
3 Planning Surveys	9
4 Other Compilation Resources	9
II LANGUAGES	11
5 STRIPS	11
6 Action Description Language (ADL)	11
7 PDDL	11
8 OPT	11
9 Darpa Agent Markup Language (DAML)	11
III FULLY AUTOMATED PLANNERS	12
10 Planner Indexes	12
10.1 BY PLANNING COMPETITION	12
10.1.1 IPC1 AT AIPS 1998	12
10.1.2 IPC2 AT AIPS 2000	12
10.1.3 IPC3 AT ICAPS 2002	13
10.1.4 IPC4 AT ICAPS 2004	13
10.1.5 IPC5 AT ICAPS 2006	14
10.2 BY PLANNING TECHNOLOGY	14
10.2.1 PARTIAL ORDER CAUSAL LINK	14
10.2.2 CONSTRAINT PROGRAMMING / CSP / IP / MLP	14
10.2.3 GRAPHPLAN	14
10.2.4 RELAXED GRAPHPLAN	14
10.2.5 HEURESTIC SEARCH, LOCAL SEARCH	14
10.2.6 MODEL CHECKING	14
10.3 BY PLANNER CAPABILITY	14

10.4 BY CONCEPTUAL LINEAGE	14
10.5 BY CODE LINEAGE	14
11 AltAlt	16
11.1 ALTALT LISP	16
11.2 ALTALT 1.0 (BUS-4.0 ENABLED)	16
12 BlackBox	16
13 BUS	17
13.1 BUS 2.0	17
13.2 BUS 3.0	17
13.3 JBUS 1.0	17
13.4 BUS 4.0	17
14 CPlan	17
14.1 CPLAN VERSION (BUS??)	17
15 CPT	18
15.1 CPT 1.0 (BUS-4.0 ENABLED)	18
16 Fast Downward	19
16.1 FAST DOWNWARD VERSION (BUS ???)	19
17 Fast Forward	20
17.1 FF 1.0	20
17.2 FF 2.2	21
17.3 FF 2.3 (BUS-4.0 ENABLED)	22
17.4 METRIC-FF 2003 (BUS-4.0 ENABLED)	23
18 GRT - Greedy Regression Tables	25
18.1 GRT 1.0	26
18.2 GRT 2.0	26
18.3 MO-GRT	26
19 HAP	26
19.1 HAP VERSION	26
19.2 HAPRC (BUS ???)	26
20 HSP - Heuristic Search Planner	27
20.1 HSP 1.0 / HSPR 1.0	27
20.2 HSP 1.1 / HSPR 1.1	27
20.3 HSP 2.0 (BUS-4.0 ENABLED)	27
20.4 HSP 2.0 R-H1MAX (BUS-4.0 ENABLED)	27
20.5 HSP 2.0 R-H1PLUS (BUS-4.0 ENABLED)	27

21 HSP*	28
21.1 HSP* 1.0 (BUS??)	28
21.1.1 PARASTAR	28
21.1.2 QP	28
21.1.3 TP4	28
21.1.4 MAXP	28
21.2 HSP* 2.0 (BUS??)	29
21.2.1 HSP_0 (ALSO CALLED TP4-04)	29
21.2.2 HSP_a	29
21.2.3 HSP_b	29
21.2.4 HSP_c	29
21.2.5 HSP_d	29
22 IPP	30
22.1 IPP 2.0	31
22.2 IPP 3.0	31
22.3 IPP 3.1	31
22.4 IPP 3.2	31
22.5 IPP 3.3 (BUS 3.0)	31
22.6 IPP 4.0 (BUS 3.0)	31
22.7 IPP 4.1 (BUS-4.0 ENABLED)	31
23 LPG	33
23.1 GPG 1.0 (APPROX. 1999)	33
23.2 LPG 1.1 (1/8/2003) (BUS-4.0 ENABLED)	34
23.3 LPG 1.2 (9/27/2003) (BUS-4.0 ENABLED)	34
23.4 LPG-TD 1.0 (08/31/2004) (BUS-4.0 ENABLED)	35
24 Marvin	36
24.1 MARVIN VERSION (BUS ???)	36
25 MetricFF	37
26 MIPS	37
26.1 MIPS 3.0 (BUS-4.0 ENABLED)	37
27 Optop	37
27.1 OPTOP 1.6 (BUS ERROR)	37
27.2 OPTOP 1.6 (BUS-4.0 ENABLED)	38
28 PbR - Planning by Rewriting	38
28.1 PbR VERSION	38
29 P-Mep	38
29.1 P-MEP VERSION	38

30	PRODIGY	39
30.1	PRODIGY 1.0	39
30.2	PRODIGY 2.0	39
30.3	NoLIMIT	39
30.4	PRODIGY 4.0 (BUS-4.0 ENABLED)	40
30.5	FLECS	42
30.6	RASPUTIN	42
31	RePOP	42
31.1	RePOP 2001	42
32	System R	43
32.1	SYSTEM R 1.1 (BUS-4.0 ENABLED)	44
33	SAPA	45
33.1	SAPA 2 (200406) (BUS-4.0 ENABLED)	45
34	SATPLAN	46
34.1	SATPLAN 1.0	46
34.2	SATPLAN 1.1	46
34.3	SATPLAN 1.2	47
34.4	BLACKBOX 2.0	47
34.5	BLACKBOX 2.5 (BUS-3.0 ENABLED) (BUS4.0 ERROR)	47
34.6	BLACKBOX 3.6B (BUS3.0 ENABLED) (BUS4.0 ERROR)	47
34.7	BLACKBOX 4.2 (BUS-4.0 ENABLED)	48
34.8	BLACKBOX 4.3 (BUS-4.0 ERROR)	49
34.9	SATPLAN 2004 (BUS-4.0 ENABLED)	50
34.10	SATPLAN 2006 (BUS-4.0 ENABLED)	50
35	SGP - Sensory Graphplan	51
35.1	SGP 1.0B (BUS3.0) (BUS-4.0 ENABLED)	52
35.2	SGP 1.0H (BUS-4.0 ENABLED)	53
36	SGPlan	54
36.1	SGPLAN 4.0 (JUNE 2004) (BUS-4.0 ENABLED)	55
36.2	SGPLAN 4.1 (AUGUST 2006) (BUS-4.0 ENABLED)	56
37	SimPlanner	56
37.1	SIMPLANNER 1.0	56
37.2	SIMPLANNER 2.0 (BUS-4.0 ENABLED)	57
37.3	SIMPLANNER 3.0 (BUS-4.0 UNCHECKED)	57
38	SNLP	58
38.1	SNLP 1.0 (BUS-4.0 ENABLED)	58
39	STAN	58
39.1	STAN 1.0 (BUS 4.0 ERROR)	58
39.2	STAN 2.3	59
39.3	STAN 3.0 (BUS4.0 ENABLED)	59

39.4 STAN 3.0s	59
39.5 STAN 4.0 (BUS-4.0 ERROR)	59
40 UCPOP	61
40.1 UCPOP 2.0	61
40.2 UCPOP 4.1 (BUS-4.0 ENABLED)	62
41 VHPOP	63
41.1 VHPOP 2.2 (BUS-4.0 ENABLED)	64
42 YAHSP - Yet Another Heuristic Search Planner	64
42.1 YAHSP VERSION	64
43 Zeno	65
43.1 ZENO VERSION	65
IV DOMAINS AND PROBLEMS	66
44 Summary by Year	66
45 Summary by Domain Name	66
46 PDDL Language Features	66
47 AIPS1998 - First International Planning Competition (IPC1)	67
47.1 MOVIE	67
47.2 GRIPPER	67
47.3 LOGISTICS	68
47.4 MYSTERY	68
47.5 MPRIME	69
47.6 GRID	69
47.7 ASSEMBLY	69
48 AIPS2000 - Second International Planning Competition (IPC2)	71
48.1 BLOCKS-WORLD	71
48.2 LOGISTICS	71
48.3 SCHEDULE	72
48.4 FREECELL	72
48.5 MICONIC	73
49 ICAPS2002 - Third International Planning Competition (IPC3)	74
49.1 DEPOTS	74
49.2 DRIVERLOG	75
49.3 ZENOTRAVEL	75
49.4 SATELLITE	76
49.5 ROVERS	77
49.6 FREECELL	78
49.7 SETTLERS	78
49.8 UMTRANSLOG-2	79

49.9	ADDITIONAL (NON-COMPETITION) DOMAINS	79
50	ICAPS2004 - Fourth International Planning Competition (IPC4)	81
50.1	AIRPORT	81
50.2	PIPESWORLD	82
50.3	PROMELA	82
50.4	PSR	83
50.5	SATELLITE	83
50.6	SETTLERS	83
50.7	UMTS	84
51	ICAPS2006 - Fifth International Planning Competition (IPC5)	85
51.1	THE TRAVELLING PURCHASER	85
51.2	OPENSTACKS	87
51.3	STORAGE	88
51.4	TRUCKS	89
51.5	PATHWAYS	91
51.6	EXTENDED ROVERS	92
51.7	EXTENDED PIPESWORLD	93
52	The FF Domain Collection (hoffmann)	94
52.1	ASSEMBLY	94
52.2	BLOCKSWORLD-3OPS	95
52.3	BLOCKSWORLD-4OPS	95
52.4	BRIEFCASEWORLD	96
52.5	FERRY	96
52.6	GRIPPER	96
52.7	HANOI	97
52.8	LOGISTICS	97
52.9	MICONIC-ADL	97
52.10	MICONIC-SIMPLE	98
52.11	MICONIC-STRIPS	99
52.12	MOVIE	99
52.13	MPRIME	99
52.14	MYSTERY	100
52.15	SCHEDULE	101
52.16	TSP	102
52.17	TYREWORLD	102
53	UCPOP Benchmarks (strict)	103
53.1	ART	103
53.2	BLOCKS-WORLD	103
53.3	BRIEFCASE	103
53.4	D1S1	103
53.5	DS	104
53.6	EIGHT	104
53.7	FERRY	104
53.8	FRIDGE	105

53.9	HOMEOWNER	105
53.10	PP-DOMAINS	105
53.11	LOGISTICS	106
53.12	MEET-PASS	106
53.13	MOLGEN	106
53.14	MONKEY	106
53.15	MONTLAKE	107
53.16	MORRIS	107
53.17	MYSTERY	107
53.18	OCCAM	108
53.19	SAFETY-DOMAIN	108
53.20	FIRE-WORLD	108
53.21	TRAINS	108
53.22	TRANSPLAN	109
53.23	TRAVEL	109
53.24	TRUCKWORLD	109
53.25	WOODSHOP	110
54	Other Domains	111
54.1	LINK CHAIN	111
54.2	SODOR	111
54.3	STORAGE TEK (STEK)	111
54.4	TEMPLATE	111
54.5	MEMON	112
54.6	PSPLIB	112
V	DOMAIN ANALYSIS TOOLS	114
55	TIM (STAN's Type Interface Module)	114
56	RIFO (Removing irrelevant facts and operators from planning problems)	114
57	GAM (IPP's Goal Agenda Manager)	114
58	XGV 1.1	114
59	PDB X.X	115
60	DISCOPLAN	115
61	VAL - Automatic Validation Tool for PDDL	115
62	RedOp - Reduced Operator Sets (now called pddlcat)	115

Part I

INTRODUCTION

This document describes, as much as possible, the set of planners, problems, and analysis tools for *Classical Planning*. This document is a work-in-progress and constructed as Mark Roberts completes his dissertation proposal – there are about 60 references scattered in text form throughout the document and another 100 or so that are sitting in the file cabinet. The current focus is on content and linking; once most of the content is in place, we plan to reorganize the document. If you would like to be notified of significant updates or changes to this document, please email MARK.

The document is a hyperlink-enabled PDF file with numerous cross-references that provide the ability to browse the file to gather information. Each planner is described in a simple, tabular and prose format that is searchable and linkable. Additionally, a single table displays all planners and information about them. The planners are cross-tabulated with entries from the problem files with known references. Each planner and problem chapter has the references for that body of work. The final annotated bibliography contains both the abstract and annotations of specific papers when they are available.

1 Acknowledgments

A small amount of this material was initially written by Clayton Daylin and Tim Fluharty. Daylin ported Bus3.0 to Java which incorporated the following 11 planners: Blackbox versions 2.5 and 3.6b, FF (Fast Forward) XXX, HSP 2.0, Prodigy 4.0, R 1.1, SGP 1.0h, SNLP 1.0, UCPOP 4.1, VHPOP 2.2. He created three files that we have borrowed from as a foundation for Bus4.0. The first file, `PlannerList.txt`, included a list of all planners Daylin knew about as of 2003. The second file, `PlannerConfigurations.txt`, provides detailed notes about how Daylin setup the planners to run in jBus. Both of these files are located in the 'daylin' subdirectory. The planner specific content of Daylin's configuration file is included in the working notes files for the respective configurations of Bus4.0. The final file Daylin, `PlannerDescriptions.txt`, provided a foundation from which this document was constructed. Daylin included details regarding:

- the level of automation
- brief description of the algorithm
- input language
- Related Planners
- Authors
- Some early or significant references
- some discussion of the heuristic and restrictions

Fluharty also provides two files that outline his work on distinguishing planner performance. Both files are located in the 'fluharty' subdirectory. The file `planner-list.html` includes basic descriptions of three planners: AltAlt 1.0, GRT 1.0, and STAN 4.0. The file

`planner-components.html` takes a closer look at the components of these three planners. In particular, Fluharty outlines the motivation, data structures, and (sometimes) pseudocode for:

- STAN's Spike [LF99],
- Nebel's RIFO [NDK97],
- GRT's Removal of Irrelevant Objects, adding negated predicates to the initial state [RV01] as well as computing state constraints (also called XOR constraints) [RV00],
- Completing incomplete Goals [RVT99]

Fluharty also mentions:

- Smith and Peot's Operator Graphs [SP96]
- STAN's wavefront search mechanism [LF99]
- DISCOPLAN's inferring state constraints [GS03]

Mark Roberts combined these documents in 2006 and maintains this version for his work in the MEPS Planning Group. Though Daylin's `PlannerDescriptions.txt` formed the foundation, that material is significantly extended for many planners.

A large part of the descriptions for the systems and problems are copied directly from their respective authors. Please note that dates of composition and authorship may be obscured by poor attribution in the document(s) – poor attribution is especially true for websites. Our goal is to properly attribute all documents whenever possible; if you feel that we are in error or there is further attribution or clarification needed, please email MARK.

2 Copyright and Colophon

This document in any form is copyright of Adele Howe and Mark Roberts, 2002-2006; all rights are reserved.

The actual \LaTeX document was created with plain old emacs and rendered using pdf \TeX Version 3.141592-1.21a-2.2 (Web2C 7.5.4) with the `verbatim`, `hyperref`, and `bibunits` packages. **DEBUG: Conversion to HTML was done using XXX.** References were managed by JabRef 2.1 available from SOURCEFORGE.

3 Planning Surveys

[RN02], chapter 11.
[GNT04]
[Liu04]
[Pla]

4 Other Compilation Resources

[Ama03]

References

- [Ama03] Rob St. Amant. Planning resources. Web Page, July 9 2003. Last accessed January 3, 2006.
- [GNT04] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning*. Morgan Kaufmann Publishers, May 2004.
- [GS03] Alfonso Gerevini and Len Schubert. Discovering state constraints for planning: Disoplan. Technical Report 811, Dept. of Computer Science, University of Rochester, Rochester, USA, September 2003.
- [LF99] D. Long and M. Fox. Efficient implementation of the plan graph in STAN. *JAIR*, 10:87–115, 1999.
- [Liu04] Donghong Liu. PLANET planning database: Online research database. Web Page, March 9 2004. Last accessed January 9, 2004.
- [NDK97] Bernhard Nebel, Yannis Dimopoulos, and Jana Koehler. Ignoring irrelevant facts and operators in plan generation. In *Proceedings of the 4th European Conference on Planning*, volume 1348 of LNAI, pages 338–350. Springer, 1997.
- [Pla] PlanSIG. Plansig: Planners and schedulers. Web Page.
- [RN02] Stuart Russell and Peter Norvig. *Introduction to Artificial Intelligence*. Prentice Hall, second edition, 2002.
- [RV00] I. Refanidis and I. Vlahavas. Exploiting state constraints in heuristic state-space planning. In *5th International Conference on Artificial Intelligence Planning and Scheduling Systems (AIPS-2000)*, Breckenridge, Colorado, April 2000.
- [RV01] I. Refanidis and I. Vlahavas. The grt planner: Backward heuristic construction in forward state-space planning. *Journal of Artificial Intelligence Research*, 15:115–161, 2001. The companion source code appendix includes the GRT Planner, AltAlt, FF, HSP, and STAN as well as a number of problems and data used to complete the paper.
- [RVT99] I. Refanidis, I. Vlahavas, and L. Tsoukalas. On determining and completing incomplete states in strips domains. In *IEEE International Conference on Information, Intelligence and Systems*, pages 289–296, Washington D.C., November 1-3 1999.
- [SP96] David E. Smith and Mark A. Peot. Suspending recursion in causal link planning. In B. Drabble, editor, *Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems (AIPS-96)*, pages 182–190. AAAI Press, 1996.

Part II

LANGUAGES

5 STRIPS

6 Action Description Language (ADL)

7 PDDL

[AIP98] [?]

8 OPT

9 Darpa Agent Markup Language (DAML)

[Unk06]

References

[AIP98] AIPS-98 International Planning Competition Committee. PDDL : the planning domain definition language. Technical report, Yale Center for Computational Vision and Control, October 1998. The AIPS-98 committee consisted of: M. Ghallab and A. Howe and C. Knoblock and D. McDermott and A. Ram and M. Veloso and D. Weld and D. Wilkins.

[Unk06] Unknown. Daml.org. Web Page, January 29 2006. Last Accessed January 9, 2007.

Part III

FULLY AUTOMATED PLANNERS

10 Planner Indexes

10.1 by planning competition

10.1.1 IPC1 at AIPS 1998

Five contestants participated in the First Planning Systems Competition:

Planner: Blackbox Authors: Henry Kautz and Bart Selman Affiliation: ATT and ATT/Cornell
Track: STRIPS Language: C

Planner: HSP Authors: Blai Bonet and Hector Geffner Affiliation: Simon Bolivar University
Track: Strips Language: C/gcc

Planner: IPP (See Section 22.5) Authors: Jana Koehler Affiliation: Freiburg University
Tracks: STRIPS, ADL Language: C/C++ (gcc/gpp)

Planner: SGP (See Section 35.1) Authors: Corin Anderson and Dan Weld Affiliation:
University of Washington Tracks: STRIPS, ADL Language: Lisp

Planner: STAN 1.0 (See Section 39.1) Authors: Derek Long and Maria Fox Affiliation:
Durham University Tracks: strips Language: Gnu C++ (g++) and Gnu C (gcc).

10.1.2 IPC2 at AIPS 2000

Participants (as listed on the website [Bac00]).

Planner: FF Author: Joerg Hoffmann Affiliation: Albert Ludwigs University Germany

Planner: GRT Authors: Ioannis Refanidis, Ioannis Vlahavas, Dimitris Vrakas Affiliation:
Aristotle University, Greece

Planner: System R Author: Fangzhen Lin Affiliation: The Hong Kong University of Science
& Technology, Hong Kong

Planner: MIPS Authors: Stefan Edelkamp, Malte Helmert Affiliation: University of Freiburg,
Germany

Planner: IPP Authors: Jana Koehler, Joerg Hoffmann, Michael Brenner Affiliation: Schindler
Lifts Ltd., Switzerland, University of Freiburg, Germany

Planner: HSP2 Authors: Hector Geffner, Blai Bonet Affiliation: Universidad Simon Boli-
var, Venezuela

Planner: PropPlan Author: Michael Fourman Affiliation: University of Edinburgh, UK

Planner: STAN 4.0 (See Section 39.5) Authors: Derek Long, Maria Fox Affiliation: Uni-
versity of Durham, UK

Planner: BlackBox Authors: Henry Kautz, Bart Selman, Yi-Cheng Huang Affiliation:
AT&T Research, Cornell University, USA

Planner: AltAlt Authors: Biplav Srivastava, Terry Zimmerman, BinhMinh Do, XuanLong
Nguyen, Zaiqing Nie, Ullas Nambiar, Romeo Sanchez Affiliation: Arizona State University,
USA

Planner: TokenPlan Authors: Yannick Meiller, Patrick Fabiani Affiliation: ONERA -
Center of Toulouse, France

Planner: BDDPlan Author: Hans-Peter Stoerr Affiliation: Dresden University of Technol-
ogy

10.1.3 IPC3 at ICAPS 2002

The Fully Automated Planners

“There were eleven competitors in this category, representing at least four distinct planning paradigms (forward search, model-checking, local search and partial order planning). Fully-automated planners accept the PDDL2.1 specifications of domain, initial state and goal and compute solutions solely on the basis of these specifications. No additional control knowledge or guidance is supplied. Fully-automated planners therefore depend on sophisticated search control heuristics and the efficient storage of alternative search branches. A popular current approach to search control is to make use of variants of the relaxed plan idea originally proposed by McDermott UNPOP and subsequently exploited by Bonet and Geffner HSP and Hoffmann FF.” [Citation Unknown]

* FF * IxTeT * LPG * MIPS * SAPA * SEMSYN * SIMPLANNER * STELLA

Planner: TP4 (part of the HSP* 1.0 suite, See Section 21.1) Authors: Patrik Haslum
Affiliation: Linköping University

* TPSYS * VHPOP

10.1.4 IPC4 at ICAPS 2004

Classical Part Sub-optimal Planners

* CRIKEY (Keith Halsey) * FAP (Guy Camilleri and Joseph Zalaket) * Fast Downward (Malte Helmert and Silvia Richter) * Fast Diagonally Downward (Malte Helmert and Silvia Richter) * LPG-TD (Alfonso Gerevini, Alessandro Saetti, Ivan Serina, and * Paolo Tonelli) * Macro-FF (Adi Botea, Markus Enzenberger, Martin Mueller, Jonathan Schaeffer) * Marvin (Andrew Coles and Amanda Smith) * Optop (Drew McDermott) * P-MEP (Javier Sanchez, Minh Tang, and Amol D. Mali) * Roadmapper (Lin Zhu and Robert Givan) * SGPlan (Yixin Chen, Chih-Wei Hsu and Benjamin W. Wah) * Tilsapa (Bharat Ranjan Kavuluri and Senthil U) * YAHSP (Vincent Vidal)

Optimal Planners

* BFHSP (Rong Zhou and Eric A. Hansen) * CPT (Vincent Vidal and Hector Geffner)

Planner: HSP*-a (part of the HSP* 2.0 suite, See Section 21.2) Authors: Patrik Haslum
Affiliation: Linköping University

* Optiplan (Menkes van der Briel and Subbarao Kambhampati) * SemSyn (Eric Parker) * SATPLAN-04 (Henry Kautz, David Rozyai, Farhad Teydaye-Saheli, Shane Neph, and Michael Lindmark)

Planner: TP4-04 (part of the HSP* 2.0 suite, See Section 21.2) Authors: Patrik Haslum
Affiliation: Linköping University

10.1.5 IPC5 at ICAPS 2006

10.2 by planning technology

10.2.1 Partial Order Causal Link

10.2.2 Constraint Programming / CSP / IP / MLP

10.2.3 Graphplan

10.2.4 Relaxed Graphplan

10.2.5 Heuristic Search, Local Search

10.2.6 Model Checking

10.3 by planner capability

10.4 by conceptual lineage

DEBUG: Here goes the conceptual family tree of the planners

10.5 by code lineage

DEBUG: Here goes the implementation family tree of the planners

Year	IPC	Planner	Version	Bus	Language	Strips	PDDL	ADL	Fluents	TIL
1992								✓		
1998										
	1	BlackBox	???				✓			
	1	HSP	???				✓			
	1	IPP	1.2	✓			✓			
	1	SGP	1.0	✓	Lisp	✓	✓			
	1	STAN	??	✓			✓			
1999										
2000										
2001		R	1.1	✓	PL	✓	✓			
2002										
2003		SGPlan			C/C++		✓		✓	
2004										
2005										
????		LPG	1.1	✓	C++		✓			
		LPG	1.2	✓	C++		✓			
		AltAlt CPP	1.0	✓			✓			
		BlackBox	4.2	✓			✓			
		FF	2.3	✓	C		✓			
		IPP	1.2	✓			✓			
		HSP	2.0	✓			✓			
		Prodigy	4.0	✓			✓			
		MIPS	3	✓			✓			
		UCPOP	4.1	✓			✓			
		SGP								
		STAN	4	✓			✓			
		SNLP	1.0	✓			✓			
		SIMPLANNER	1.0	✓			✓			
		VHPOP	2.2	✓	C++		✓			
		LPG-td					✓			

Table 2: Planner Summary by Year

11 AltAlt

Authors	XuanLong Nguyen, Subbarao Kambhampati, Romeo Sanchez Nigenda
Links	ALTALT WEBSITE REPOSITORY LINK
RelatedTo	STAN 3.0 (Section 39.3), HSP-r (Section 20)
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

11.1 AltAlt Lisp

Description	This is the original version of the planner, written in LISP. The authors ported this version to C for the competition. DEBUG: Locate this citation; it is probably [?]
-------------	--

11.2 AltAlt 1.0 (BUS-4.0 Enabled)

Description	[?] outlines the basic configuration of this version.
Setup	<pre>planner.name = AltAlt planner.version = cpp1.0 planner.dirname = altalt-1.0 planner.fullname = altalt-1.0 planner.encoding = Hybrid working.directory = {%system.directory%}/{%planner.dirname%}/ exec.command = altalt {%DOMAIN_PATH%} {%PROBLEM_PATH%} # NOTE: AltAlt writes non-error output to both standard output and standard # error! success.test.command = grep -q 'Length of plan' {%OUTPUT_PATH%}</pre>
Notes	NO NOTES IN THIS FILE
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

12 BlackBox

BLACKBOX is the name of the set of planners that are in the middle portion of the SATPLAN lineage. This planner is covered in detail in Subsection 34.

13 BUS

Authors	Adele Howe, Eric Dahlman, Clayton Daylin, Mark Roberts	
Links	BUS WEBSITE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here.	
Description	Detailed description goes here.	

13.1 BUS 2.0

Description Best Described in [Citation Unknown].

13.2 BUS 3.0

Description Best Described in in [HDH⁺99].

13.3 jBUS 1.0

Description Best Described in [Citation Unknown].

13.4 BUS 4.0

Description The most recent version, Bus4.0 contains the code used for the papers from 2005 to the present. [Citation Unknown].

References

[HDH⁺99] Adele Howe, Eric Dahlman, C. Hansen, A. vonMayrhauser, and M. Scheetz. Exploiting competitive planner performance. In *Proc. of ECP-99*, Durham, England, September 1999.

14 CPlan

Authors		
Links	WEBSITE UNAVAILABLE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here.	
Description	Detailed description goes here.	

14.1 CPlan VERSION (BUS??)

Description Best Described in [VBC99].

References

- [VBC99] P. Van Beek and X. Chen. Cplan: a constraint programming approach to planning. In *Proc. National Conference on Artificial Intelligence*, pages 585–590. AAAI Press/MIT Press, 1999.

15 CPT

Authors	Hector Geffner, Vincent Vidal
Links	CPT WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

15.1 CPT 1.0 (BUS-4.0 Enabled)

Description	Best Described in [?].
Setup	<pre>planner.name = CPT planner.version = 1.0 planner.dirname = cpt-1.0 planner.fullname = cpt-1.0 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = cpt-1.0.linux.x86 -s 6 10 -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'Length :' {%OUTPUT_PATH%}</pre>
Notes	<p>2005-10-04 Claire will not run in 256M. Will need to up the memeory before I can run this planner.</p> <p>2005-10-05 Claire needs more memory still. The problem is that the default of 2⁷ stack and 2¹² heap are too large for 768M limit. Need to set to 2⁶ and 2¹⁰ respectively. However, then we are providing Claire with allowing 512M. Is there a way to specify "use as much as you like until you barf"? There doesn't seem to be, but finding a manual and/or understanding the code is quite challenging....</p>
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

16 Fast Downward

Authors	Malte Helmert and Silvia Richter
Links	WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	Fast Forward
Summary	<p>“Fast Downward is a classical planning system based on heuristic search. It can deal with general deterministic planning problems encoded in the propositional fragment of PDDL2.2, including advanced features like ADL conditions and effects and derived predicates (axioms). Like other well-known planners such as HSP and FF, Fast Downward is a progression planner, searching the space of world states of a planning task in the forward direction. However, unlike other PDDL planning systems, Fast Downward does not use the propositional PDDL representation of a planning task directly. Instead, the input is first translated into an alternative representation called multi-valued planning tasks, which makes many of the implicit constraints of a propositional planning task explicit. Exploiting this alternative representation, Fast Downward uses hierarchical decompositions of planning tasks for computing its heuristic function, called the causal graph heuristic, which is very different from traditional HSP-like heuristics based on ignoring negative interactions of operators. In this article, we give a full account of Fast Downward’s approach to solving multi-valued planning tasks. We extend our earlier discussion of the causal graph heuristic to tasks involving axioms and conditional effects and present some novel techniques for search control that are used within Fast Downward’s best-first search algorithm: preferred operators transfer the idea of helpful actions from local search to global best-first search, deferred evaluation of heuristic functions mitigates the negative effect of large branching factors on search performance, and multi-heuristic best-first search combines several heuristic evaluation functions within a single search algorithm in an orthogonal way. We also describe efficient data structures for fast state expansion (successor generators and axiom evaluators) and present a new non-heuristic search algorithm called focused iterative-broadening search, which utilizes the information encoded in causal graphs in a novel way. Fast Downward has proven remarkably successful: It won the “classical” (i.e., propositional, non-optimising) track of the 4th International Planning Competition at ICAPS 2004, following in the footsteps of planners such as FF and LPG. Our experiments show that it also performs very well on the benchmarks of the earlier planning competitions and provide some insights about the usefulness of the new search enhancements.” [Hel06]</p> <p>Summary description (400 words) goes here.</p>
Description	Detailed description goes here.

16.1 Fast Downward VERSION (BUS ???)

Description Best Described in [Citation Unknown].

References

[Hel06] Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.

17 Fast Forward

Authors	Joerg Hoffmann, Bernard Nebel		
Links	FF WEBSITE	METRICFF WEBSITE	REPOSITORY LINK
RelatedTo			
Summary	Summary description (400 words) goes here.		
Description	Detailed description goes here.		

17.1 FF 1.0

Description	Best Described in [?].
Setup	<pre> planner.name = FF planner.version = 2.3 planner.dirname = ff-2.3 planner.fullname = ff-2.3 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = ff -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'found legal plan' {%OUTPUT_PATH%} </pre>
Notes	<p>200307 Daylin</p> <p>The directory argument is unnecessary if absolute filepaths are used for the domain and problem arguments. (In "main.c" the directory string is simply prepended to the two filenames.)</p>
Feature Summary	<pre> plan-ipc = IPC3 plan-year = 2002 plan-age = 5 plan-programmers = 3 plan-version = 3 plan-impl-lang = C plan-tech-rgp = true plan-tech-hs = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-alg-hc-enf = true plan-pre-ippadl2str = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-rep-state = true plan-rep-clsdwrlld = true plan-heu-hplus-add = true plan-agda-lstcst = true plan-sol-para = true </pre>

17.2 FF 2.2

Description	Best Described in [?].
Setup	<pre> planner.name = FF planner.version = 2.3 planner.dirname = ff-2.3 planner.fullname = ff-2.3 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = ff -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'found legal plan' {%OUTPUT_PATH%} </pre>
Notes	<p>200307 Daylin</p> <p>The directory argument is unnecessary if absolute filepaths are used for the domain and problem arguments. (In "main.c" the directory string is simply prepended to the two filenames.)</p>
Feature Summary	<pre> plan-ipc = IPC3 plan-year = 2002 plan-age = 5 plan-programmers = 3 plan-version = 3 plan-impl-lang = C plan-tech-rgp = true plan-tech-hs = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-alg-hc-enf = true plan-pre-ippadl2str = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-rep-state = true plan-rep-clsdwrlld = true plan-heu-hplus-add = true plan-agda-lstcst = true plan-sol-para = true </pre>

17.3 FF 2.3 (BUS-4.0 Enabled)

Description	Best Described in [?].
Setup	<pre> planner.name = FF planner.version = 2.3 planner.dirname = ff-2.3 planner.fullname = ff-2.3 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = ff -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'found legal plan' {%OUTPUT_PATH%} </pre>
Notes	<p>200307 Daylin</p> <p>The directory argument is unnecessary if absolute filepaths are used for the domain and problem arguments. (In "main.c" the directory string is simply prepended to the two filenames.)</p>
Feature Summary	<pre> plan-ipc = IPC3 plan-year = 2002 plan-age = 5 plan-programmers = 3 plan-version = 3 plan-impl-lang = C plan-tech-rgp = true plan-tech-hs = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-alg-hc-enf = true plan-pre-ippadl2str = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-rep-state = true plan-rep-clsdwrlld = true plan-heu-hplus-add = true plan-agda-lstcst = true plan-sol-para = true </pre>

17.4 Metric-FF 2003 (BUS-4.0 Enabled)

Description	Best Described in [Hof03].
Setup	<pre>planner.name = Metric-FF planner.version = 2002 planner.dirname = metric-ff_2002 planner.fullname = metric-ff_2002 planner.encoding = RGP working.directory = {%system.directory%}/metric-ff_2002 exec.command = ff -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'found legal plan' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = IPC3 plan-year = 2002 plan-age = 3 plan-programmers = 3 plan-version = 4 plan-impl-lang = C plan-tech-rgp = true plan-tech-hs = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-alg-hc-enf = true plan-pre-ippadl2str = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-lang-fluents = true plan-lang-pddl21 = true plan-rep-state = true plan-rep-clsdwrlld = true plan-heu-hplus-add = true plan-agda-lstcst = true plan-sol-para = true</pre>

References

- [Hof03] J. Hoffmann. The Metric-FF planning system: Translating “ignoring delete lists” to numeric state variables. *JAIR*, 20:291–341, 2003.
-

18 GRT - Greedy Regression Tables

Authors	
Links	GRT WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	<p>“The GRT planner is a domain-independent heuristic planner, which adopts the pure STRIPS representation and searches forward in the space of the states. The planner has initially been inspired by the ASP/HSP planner, but it has been differentiated in several ways.</p> <p>GRT solves planning problems in two phases: the pre-processing phase and the search phase. The main idea of the planner is to compute off-line, in the pre-processing phase, estimates for the distances between the domains facts and the goals. The word ‘distance’ refers to the number of goal regression levels needed to achieve a specific fact. This information is stored in a table, which is indexed by the facts of the domain. We call this table the Greedy Regression Table, which the acronym GRT comes from.</p> <p>In order to produce better estimates, GRT introduces the notion of related facts in the goal regression process. These are facts that have been achieved either by the same or by subsequent actions, without the last action deleting the firstly achieved facts. The cost for achieving a set of un-related facts simultaneously is the sum of their individual costs, while the cost for achieving a set of related facts is the cost of the last achieved fact.</p> <p>The search phase consists of a simple best-first search strategy. Based on the distances of the individual facts from the goals, and the information about their relations, GRT obtains estimates for the distances between the intermediate states and the goals, which are used to guide the search.</p> <p>The original version of GRT participated at the AIPS-2000 with very promising results. Actually, GRT has not gained any prize, but most prizes in the domain-independent track of the competition were given to similar planners (FF and HSP-II).” [Ref03]</p>
References	<p>2003 I. Refanidis and I. Vlahavas, “Multiobjective Heuristic State-Space Planning”, <i>Artificial Intelligence Journal</i>, vol 145/1-2 pp 1-32, 2003</p> <p>I. Refanidis, I. Vlahavas and K. Paparrizos. Resource Allocation for Crisis Management using Planning. Accepted for presentation at the 15th National Conference of the Hellenic Operation Research Society, Tripoli, 2002</p> <p>I. Refanidis and I. Vlahavas, “The MO-GRT System: Heuristic Planning with Multiple Criteria”, <i>AIPS-02 Workshop on Planning and Scheduling with Multiple Criteria</i>, Toulouse, France, 2002</p> <p>I. Stamelos and I. Refanidis, Decision Making Based On Past Problems Cases, 2nd Hellenic Conference for Artificial Intelligence, April 2002, Thessaloniki, Springer-Verlag, pp. 42-53</p> <p>I. Refanidis and I. Vlahavas, “The GRT Planner”, <i>AI-Magazine</i>, Fall 2001</p> <p>D. Vrakas, I. Refanidis and I. Vlahavas, “Parallel Planning via the Distribution of Operators”, <i>Journal of Experimental and Theoretical Artificial Intelligence</i>, to appear</p> <p>I. Refanidis and I. Vlahavas, “A Framework for Multi-Criteria Plan Evaluation in Heuristic State-Space Planning”, to be presented in <i>IJCAI-01 workshop on Planning with Resources</i>, Seattle, Washington, August 2001</p> <p>I. Refanidis and I. Vlahavas, “The GRT Planner: New Results”, In <i>ECAI-00 PostWorkshop Proceedings in Local Search Techniques for Planning and Scheduling</i>, Springer, LNAI 2148, 120-138</p> <p>I. Refanidis, N. Bassiliades and I. Vlahavas, “AI Planning in Transportation Logistics”, 17th International Logistics Congress, Thessaloniki, October 2001, to be presented</p> <p>D. Vrakas, I. Refanidis and I. Vlahavas, “An Operator Distribution Method for Parallel Planning”, <i>AAAI workshop on Parallel and Distributed Search for reasoning</i>, Texas, 2000</p> <p>I. Refanidis and I. Vlahavas, “Heuristic Planning with Resources”, 14th European Conference on Artificial Intelligence, Berlin, August 2000.</p> <p>I. Refanidis and I. Vlahavas, “Exploiting State Constraints in Heuristic State-Space Planning”, 5th International Conference on Artificial Intelligence Planning and Scheduling Systems (AIPS-2000), Breckenridge, Colorado, USA, April 2000</p> <p>I. Refanidis and I. Vlahavas, “A Heuristic Based Approach to Planning in Strips Domains”, in <i>Advances in Informatics</i>, ed. by D.I. Fotiadis and S.D. Nikolopoulos, World Scientific, April 2000</p> <p>D. Vrakas, I. Refanidis, F. Milcent and I. Vlahavas, “On the parallelization of Greedy Regression Tables”, 18th UK Planning and Scheduling SIG meeting. Manchester, UK,</p>

18.1 GRT 1.0

Description The strongest description of the planner is found in the 2001 JAIR article [RV01].

18.2 GRT 2.0

Description Best Described in [Citation Unknown].

18.3 MO-GRT

Description Best Described in [Citation Unknown].

References

[Ref03] Ioannis Refanidis. Grt planner - home page. Web Page, June 2003. Last accessed January 3, 2007.

[RV01] I. Refanidis and I. Vlahavas. The grt planner: Backward heuristic construction in forward state-space planning. *Journal of Artificial Intelligence Research*, 15:115–161, 2001. The companion source code appendix includes the GRT Planner, AltAlt, FF, HSP, and STAN as well as a number of problems and data used to complete the paper.

19 HAP

Authors	
Links	HAP WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here. [VTBV03]

19.1 HAP VERSION

Description Best Described in [VTBV05].

19.2 HAPrc (BUS ???)

Description Best Described in [VTBV05].

References

- [VTBV03] D. Vrakas, G. Tsoumakas, N. Bassiliades, and I. Vlahavas. Learning rules for adaptive planning. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS03)*, pages 82–91, Trento, Italy, June 9-13 2003.
- [VTBV05] D. Vrakas, G. Tsoumakas, N. Bassiliades, and I. Vlahavas. *Intelligent Techniques for Planning*, chapter Machine Learning for Adaptive Planning, pages 90–120. Idea Group, 2005.
-

20 HSP - Heuristic Search Planner

Authors	Blai Bonet, Hector Geffner
Links	HSP WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

20.1 HSP 1.0 / HSPr 1.0

Description This version was the original version used in IPC1 (See Section ??). Best Described in [?].

20.2 HSP 1.1 / HSPr 1.1

Description Best Described in [Citation Unknown].The authors write that “This release, 1.1, just changes the implementation of the heuristic function by making it significantly faster. As a result, the planner runs several times faster than before. [BG00]”

20.3 HSP 2.0 (BUS-4.0 Enabled)

Description Best Described in [Citation Unknown].

20.4 HSP 2.0 r-H1Max (BUS-4.0 Enabled)

Description Best Described in [Citation Unknown].

20.5 HSP 2.0 r-H1Plus (BUS-4.0 Enabled)

Description Best Described in [Citation Unknown].

References

- [BG00] Blai Bonet and Hector Geffner. *Heuristic Search Planner 1.1 README*, June 21 2000. Included with the distribution for HSP 1.1.
-

21 HSP*

Authors	Patrick Haslam and Hector Geffner
Links	HSPS WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.
References	* P. Haslum Improving Heuristics Through Search. European Conference on Artificial Intelligence, 2004 (short paper). * P. Haslum TP4'04 and hsp*a. In 4th IPC systems descriptions booklet (available as a whole from http://ipc.icaps-conference.org). * P. Haslum, H. Geffner, Heuristic Planning with Time and Resources. European Conference on Planning, 2001. * P. Haslum, H. Geffner, Admissible Heuristics for Optimal Planning. Proc. International Conference on AI Planning and Scheduling, 2000. * P. Haslum, U. Scholz Domain Knowledge in Planning: Representation and Use. ICAPS Workshop on PDDL, 2003. [Has04]

21.1 HSP* 1.0 (BUS??)

Description Best Described in [Citation Unknown].

21.1.1 Parastar

Description Best Described in [Citation Unknown].

21.1.2 QP

Description Best Described in [Citation Unknown].

21.1.3 TP4

Description Best Described in [Citation Unknown].

21.1.4 Maxp

Description Best Described in [Citation Unknown].

21.2 HSP* 2.0 (BUS??)

Description Best Described in [Citation Unknown].

21.2.1 HSP_0 (also called TP4-04)

Description Best Described in [Citation Unknown].

21.2.2 HSP_a

Description Best Described in [Citation Unknown].

21.2.3 HSP_b

Description Best Described in [Citation Unknown].

21.2.4 HSP_c

Description Best Described in [Citation Unknown].

21.2.5 HSP_d

Description Best Described in [Citation Unknown].

References

[Has04] Patrik Haslum. Hsp* homepage. Web Page, July 2004. Last accessed January 9, 2007.

22 IPP

Authors	
Links	IPP WEBSITE REPOSITORY LINK
RelatedTo	
Summary	“IPP [?] extends the Graphplan [?] algorithm to accept a richer plan description language. In its early versions, this language was a subset of ADL that extends the STRIPS formalism of Graphplan to allow for conditional and universally quantified effects in operators. Until version 4.0, negation was handled via the introduction of new predicates for the negated preconditions and corresponding mutual exclusion rules; subsequent versions handle it directly [?]. We used the AIPS98 version of IPP as well as the later 4.0 version.” [HD02]
Description	<p>“After the Dagstuhl Workshop on Control of Search in Planning, Jana Koehler decided to develop her own planning system as the topic of her habilitation thesis. It began with an extension of GRAPHPLAN to the ADL planning language handling conditional effects and more expressive preconditions. The first algorithm prototypes were discussed with Bernhard Nebel and Yannis Dimopoulos and soon the IPP planning system began to emerge. Bernhard Nebel later contributed the RIFO strategy to IPP. In march 1997 three students, Joerg Hoffmann, Frank Rittinger, and Michael Brenner, joined the team and implemented the first prototype of IPP as well as the subsequent revised versions. Version 3.3 won the ADL track of the first planning systems competition. In 1998, a first investigation into metric planning/resource-constrained planning started and experimental algorithms were implemented. At the same time, the whole IPP system was reimplemented, which required more effort than expected. Joerg Hoffmann took mainly care of this. Andreas Schoen and Dietmar Jaeckle joined the team temporarily in 1999 and contributed to the reimplementation resulting in IPP version 4.0. To enter the Planning Competition 2000, RIFO was integrated into IPP again and several bugs were fixed resulting in IPP 4.1. Between July 1997 and December 1998 IPP source code has been downloaded more than 200 times and the system has been used in several other projects (see e.g. the AIPS 2000 proceedings for references). IPP was also used in two commercial environments: at Schindler for rapid prototyping of elevator domain models (with our permission) and at Celcorp as a source for their own planning software development (without our permission).” [Koe06]</p> <p>The original extensions to the planning graph to handle ADL [KNHD97a] with a more detailed analysis in [KNHD97b].</p> <p>The original RIFO paper [NDK97] extended further in [Koe99].</p> <p>The initial GAM paper [Koe98] extended in the technical report [KH98] and incorporating a theoretical framework [?].</p> <p>The subset query engine for IPP is discussed [HK99].</p>
References	<p>IPP with Resource Constraints</p> <p>* J. Koehler: Planning under Resource Constraints, ECAI-98, pages 489-493. This paper describes an attempt of defining a useful representation language to handle resource information in planning problems and gives an outline of the algorithm that is partially implemented in RIPP 0.1. It has inspired some more work into this direction, see the AAAI-99 and IJCAI-99 proceedings.</p> <p>* J. Koehler: Metric Planning using Planning Graphs - A First Investigation. The Technical Report 127/99 gives a clean definition of a sound and complete backward searching algorithm to handle simple resource effects in planning graphs. It has never been implemented in IPP.</p>
References	<p>Publications about IPP 4.0</p> <p>* J. Koehler: Handling of Conditional Effects and Negative Goals in IPP. The Technical Report 128/99 describes how the graph is built implicitly by generating only a single action and fact layer and how the backward search algorithm works.</p> <p>* J. Koehler, J. Hoffmann: Handling of Inertia in a Planning System. The Technical Report 122/99 describes IPP’s approach to generate a compact and normalized instantiation of general ADL operators.</p>

22.1 IPP 2.0

Description Best Described in [KNHD97a].

22.2 IPP 3.0

Description Best Described in [KNHD97a].

22.3 IPP 3.1

Description Best Described in [KNHD97a].

22.4 IPP 3.2

Description Best Described in [KNHD97a].

22.5 IPP 3.3 (BUS 3.0)

This is the version used in IPC1 (Section 10.1.1) Description Best Described in [KNHD97a].

22.6 IPP 4.0 (BUS 3.0)

Description Best Described in [KNHD97a].

22.7 IPP 4.1 (BUS-4.0 Enabled)

Description Best Described in [KNHD97a].

References

- [HD02] Adele Howe and Eric Dahlman. A critical assessment of benchmark comparison in planning. *JAIR*, 17:1–33, July 2002.
- [HK99] J. Hoffmann and J. Koehler. A new method to query and index sets. In *Proc. of IJCAI-99*, 1999.
- [KH98] Jana Koehler and Joerg Hoffmann. Planning with goal agendas. Technical report, Institute for Computer Science, Albert Ludwigs University, 1998.
- [KNHD97a] Jana Koehler, Bernhard Nebel, Joerg Hoffmann, and Yannis Dimopoulos. Extending planning graphs to an ADL subset. In *Proc. ECP-97*, pages 273–285, London, UK, 1997. Springer-Verlag.

- [KNHD97b] Jana Koehler, Bernhard Nebel, Joerg Hoffmann, and Yannis Dimopoulos. Extending planning graphs to an ADL subset. Technical Report 88/97, Institute for Computer Science, Albert Ludwigs University, 1997.
- [Koe98] J. Koehler. Solving complex planning tasks through extraction of subproblems. In *Proc. of AIPS-98*, pages 62–69. AAAI Press, 1998.
- [Koe99] J. Koehler. RIFO within IPP. Technical Report 126/99, Institute for Computer Science, Albert Ludwigs University, Am Flughafen 17, 79110 Freiburg, Germany, 1999.
- [Koe06] Jana Koehler. IPP homepage. Web Page, October 15 2006. Last accessed September 11, 2007.
- [NDK97] Bernhard Nebel, Yannis Dimopoulos, and Jana Koehler. Ignoring irrelevant facts and operators in plan generation. In *Proceedings of the 4th European Conference on Planning*, volume 1348 of LNAI, pages 338–350. Springer, 1997.
-

23 LPG

Authors	
Links	LPG WEBSITE REPOSITORY LINK
RelatedTo	
Input Language(s)	PDDL (LPG 1.1), PDDL 2.1 (LPG 1.2), PDDL 2.2 (LPG-td)
Summary	Summary description (400 words) goes here. Algorithm - Local Search on Planning Graphs [?].
Description	Detailed description goes here. Original paper was [GS99]. Examined the role of systematic or stochastic search [GS00] [GS01], [GS03], [GSSS03] LPG-td: [GSS03],[GSS04], [GSSP04], [GSS06]
References	<p>New: Alfonso Gerevini, Alessandro Saetti, Ivan Serina, Paolo Toninelli, "Fast Planning in Domains with Derived Predicates: An Approach Based on Rule-Action Graphs and Local Search", to appear in Proceedings of the Twentieth American National Conference on Artificial Intelligence (AAAI-05), Pittsburgh, USA, 2005.</p> <p>New: Alfonso Gerevini, Alessandro Saetti, Ivan Serina, "Integrating Planning and Temporal Reasoning for Domains with Durations and Time Windows Planning with Numerical Expressions in LPG", in Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI-05), Edinburgh, Scotland, UK, 2005.</p> <p>New: Alfonso Gerevini, Alessandro Saetti, Ivan Serina, Paolo Toninelli, "Planning with Derived Predicates through Rule-Action Graphs and Relaxed-Plan Heuristics", R.T. 2005-01-40, Universit degli Studi di Brescia, Dipartimento di Elettronica per l Automazione. Brescia, Italy. 2005.</p> <p>Alfonso Gerevini, Alessandro Saetti, Ivan Serina, "Planning with Numerical Expressions in LPG", in Proceedings of the 16th European Conference on Artificial Intelligence (2004), Valencia, Spain, 2004.</p> <p>Alfonso gerevini, Alessandro Saetti, Ivan Serina, "An Empirical Analysis of Some Heuristic Features for Local Search in LPG", in Proceedings of the 14th International Conference on Automated Planning and Scheduling (ICAPS-04), AAAI Press, 2004.</p> <p>Alfonso Gerevini, Alessandro Saetti, Ivan Serina "On Managing Temporal Information for Handling Durative Actions in LPG", in A. Cappelli, F. Turini (eds.), AI*IA 2003: Advances in Artificial Intelligence. LNAI 2829, pages 91-104, Springer-Verlag Berlin Heidelberg 2003.</p>

23.1 GPG 1.0 (approx. 1999)

Description	Best Described in [GS99]. The date for this planner is somewhere in early 1999 judging from the publication date for AAAI-1999. This prototype implementation is built on top of IPP's data structures and implements the Action Graph, a subgraph of the full relaxed planning graph. From what I can tell, it is built on top of IPP 3.3 22.5 from the first competition.
-------------	---

23.2 LPG 1.1 (1/8/2003) (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown]. This appears to be the actual version that was used in IPC3. The README file for the planner reveals a date of Jan 8, 2003.
Setup	<pre> planner.name = LPG planner.version = 1.1 planner.dirname = lpg-1.1 planner.fullname = lpg-1.1 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = lpg -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} -n 1 -noout -same_objects success.test.command = grep -q 'solution found' {%OUTPUT_PATH%} </pre>
Notes	No working notes for this planner.
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

23.3 LPG 1.2 (9/27/2003) (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown]. This appears to be the version used subsequent to the IPC3 version; it appears to fix minor bugs and issues to improve the overall ratio of solved to attempted problems from 87% (428 / 508) to 94.4% (468/508). The README file for the planner reveals a date of June 2003 but the website lists the date as September 2003[Citation Unknown].
Setup	<pre> planner.name = LPG planner.version = 1.2 planner.dirname = lpg-1.2 planner.fullname = lpg-1.2 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%}/LPG exec.command = lpg -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} -n 1 -noout -same_objects success.test.command = grep -q 'solution found' {%OUTPUT_PATH%} </pre>
Notes	No working notes for this planner.
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

23.4 LPG-td 1.0 (08/31/2004) (BUS-4.0 Enabled)

Description	Best Described in [GSS06]. The name of the planner stands for L ocal search on P lanning G raphs with T imed initial literals and D erived Predicates. The README file lists a date of August 2004 (time stamped on the 31st) while the website lists June 2004[Citation Unknown].
Setup	<pre>planner.name = LPG-TD planner.version = 1.0 planner.dirname = lpg-td-1.0 planner.fullname = lpg-td-1.0 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%}/LPG-td-1.0/ exec.command = lpg-td-1.0 -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} -n 1 -noout success.test.command = grep -q 'solution found' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = IPC4 plan-year = 2004 plan-age = 5 plan-programmers = 11 plan-version = 4 plan-impl-lang = C plan-tech-rgp = true plan-tech-hs = true plan-srch-direction = UNDEF plan-alg-wastar-w = UNKNOWN plan-alg-hc = true plan-pre-rio = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-lang-fluents = true plan-lang-pddl22 = true plan-lang-til = true plan-lang-dpred = true plan-rep-prop = true plan-rep-rgp = true plan-rep-clsdwrlld = true plan-sol-para = true</pre>

References

- [GS99] Alfonso Gerevini and Ivan Serina. Fast planning through greedy action graphs. In *Proceedings of Sixteenth National Conference of Artificial Intelligence (AAAI-99)*, Orlando, Florida, USA, 1999. AAAI-MIT Press.
- [GS00] Alfonso Gerevini and Ivan Serina. Fast plan adaptation through planning graphs: Local and systematic search techniques. In *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems (AIPS-00)*, Breckenridge, Colorado, USA, 2000. AAAI Press.

- [GS01] Alfonso Gerevini and Ivan Serina. Lagrange multipliers for local search on planning graphs. In A. Nareyek, editor, *Proceedings of the ECAI workshop on Local Search for Planning and Scheduling*, number Lecture Notes in Artificial Intelligence 2148. Springer-Verlag, 2001.
- [GS03] Alfonso Gerevini and Ivan Serina. Planning as propositional CSP: from Walksat to local search for action graphs. *CONSTRAINTS*, 8:389–413, 2003.
- [GSS03] A. Gerevini, A. Saetti, and I. Serina. Planning through stochastic local search and temporal action graphs in LPG. *JAIR*, 20:239–290, 2003.
- [GSS04] Alfonso Gerevini, Alessandro Saetti, and Ivan Serina. Planning in PDDL2.2 domains with LPG-TD. In *Proc. of the 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04), International Planning Competition abstract*, Whistler, Canada, 2004.
- [GSS06] Alfonso Gerevini, Alessandro Saetti, and Ivan Serina. An approach to temporal planning and scheduling in domains with predicatable exogenous events. *JAIR*, 25:187–231, 2006.
- [GSSP04] Alfonso Gerevini, Alessandro Saetti, Ivan Serina, and PaoloToninelli. LPG-TD: a fully automated planner for PDDL2.2 domains. In *Proc. of the 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04) International Planning Competition abstracts*, Whistler, Canada, 2004.
- [GSSS03] Alfonso Gerevini, Ivan Serina, Alessandro Saetti, and Sergio Spinoni. Local search techniques for temporal planning in LPG. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS-03)*, Trento, Italy, 2003. AAAI Press.

24 Marvin

Authors	
Links	MARVIN WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

24.1 Marvin VERSION (BUS ???)

Description Best Described in [CS07].

References

- [CS07] A. I. Coles and A. J. Smith. Marvin: A heuristic search planner with online macro-action learning. *JAIR*, 28:119–156, 2007.

25 MetricFF

This planner is covered in detail in Subsection ??.

26 MIPS

Authors		
Links	WEBSITE UNAVAILABLE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here. [?]	
Description	Detailed description goes here.	

26.1 MIPS 3.0 (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown].
Setup	<pre>planner.name = MIPS planner.version = 3 planner.dirname = mips-3 planner.fullname = mips-3 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = mips {%DOMAIN_PATH%} {%PROBLEM_PATH%} success.test.command = grep -q 'Expansions' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

27 Optop

Authors		
Links	OPTOP WEBSITE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here. PDDL 2.2. [McD05]	
Description	Detailed description goes here.	

27.1 Optop 1.6 (BUS Error)

Description	Best Described in [Citation Unknown]. An early version that we could not get compiled on our system. Dr. McDermott posted a new version that we installed.
Notes	

27.2 Optop 1.6 (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown].
Setup	
Notes	

References

[McD05] Drew McDermott. *The Opt and Optop API*. Yale Computer Science Department, 0.9 edition, November 2005.

28 PbR - Planning by Rewriting

Authors	
Links	PBR WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

28.1 PbR VERSION

Description	Best Described in [AK01]. Look in “meps/planners/archive/random/pbr” to see if this the actual implementation of PbR
-------------	--

References

[AK01] J.L. Ambite and C.A. Knoblock. Planning by rewriting. *JAIR*, 15:207–261, 2001.

29 P-Mep

Authors	
Links	WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

29.1 P-Mep VERSION

Description	Best Described in [Citation Unknown].
-------------	---------------------------------------

References

30 PRODIGY

Authors		
Links	PRODIGY WEBSITE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here. The best overviews of all versions are given in [?] and [VCP+95]. The manual [Citation Unknown] appears to be a good source of inner mechanics, but the version I have is unreadable beyond page 6.	
Description	Detailed description goes here.	

30.1 Prodigy 1.0

Description Best Described in [VCP+95].

30.2 Prodigy 2.0

Description Best Described in [VCP+95].

30.3 NoLimit

Description Best Described in [VCP+95].

30.4 Prodigy 4.0 (BUS-4.0 Enabled)

Description	Best Described in [VCP+95] and [FB05].
Setup	<pre>planner.name = PRODIGY planner.version = 4.0 planner.dirname = prodigy-4.0 planner.fullname = prodigy-4.0 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = {%lisp%} -L "working/driver" -e '(do-pddl :domain-filepath "{%DOMAIN_PATH%}" :problem-filepath "{%PROBLEM_PATH%}")' success.test.command = grep -q 'Solution' {%OUTPUT_PATH%}</pre>
Feature Summary	<pre>plan-ipc = NA plan-year = 1992 plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = LISP plan-tech-tocl = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-alg-dfs = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-rep-tocl = true plan-rep-state = true plan-rep-clsdwrlld = true plan-agda-contrule = true plan-sol-serial = true</pre>

20070108 MCR

This Planner was part of the Bus3.0 system. The configuration for Bus3.0 was:

```
(define-planner ("Prodigy" "4.0")
:time-intercept -10.4526
:time-inits-coefficient 4.6442
:time-objects-coefficient -4.7509
:time-goals-coefficient 28.7998
:time-operators-coefficient 0.0
:time-predicates-coefficient -12.9877

:goals-intercept 0.4478
:goals-slope -0.0308
:inits-intercept 0.5172
:inits-slope -0.0008
:objects-intercept 0.2021
:objects-slope 0.0051
:operators-intercept 0.2840
:operators-slope 0.0368
:predicates-intercept 0.3215
:predicates-slope 0.0191
:probability-divisor 5
:exec-string (concatenate 'string
"exec /usr/local/allegro5.0.1/lisp -I "
*current-planner-directory*
"working/prodigy.dxl -e '(progn (in-package :pddl) (load \"/s/chopin/a/grad/dahlman/meps/planners/bus3fin
:success-test-string "exec /usr/xpg4/bin/grep -q \"Solution\" /tmp/prodigy.out"
:dump-file-name "/tmp/prodigy.out" )
```

200307 Daylin

The converter from PDDL syntax to Prodigy syntax (in "working/pddl2prodigy.lisp") requires that the domain and path files end in ".pddl".

I renamed "load-prodigy" to "load-prodigy-compiled", to eliminate a naming conflict.

Driver file: "working/driver.lisp"

Compiling:

See "prodigy-4.0/working/documents/installing.txt" for instructions.

NOTE: One way to determine the MACHINE-TYPE, LISP-IMPLEMENTATION-VERSION, and SOFTWARE-TYPE is to execute the function "excl:dribble-bug". (Substrings of those fields are needed for specifying the Lisp and OS versions.)

NOTE: If you skip the step "(load-source)", compilation will succeed.

However, macros will not be expanded properly, and you will encounter errors when running prodigy.

NOTE: Just leave *load-prodigy-immediately* set to 'nil'.

To compile the driver:

- 1) delete *.fasl in "prodigy-4.0/working" directory
- 2) (load "working/loader")
- 3) (compile-file "working/loader.lisp")
- 4) (load-prodigy-compiled)
- 5) (load "working/pddl2prodigy")
- 6) (compile-file "working/pddl2prodigy.lisp")
- 7) (load "working/driver")
- 8) (compile-file "working/driver.lisp")

20030626 Daylin

Prodigy is compiled to run only if the working directory is the Prodigy installation directory. This allows the system to be moved to a different directory without requiring recompilation.

30.5 FLECS

Description Best Described in [FB05].

30.6 RASPUTIN

Description Best Described in [FB05].

References

- [FB05] Eugene Fink and Jim Blythe. Prodigy bidirectional planning. *Journal of Experimental and Theoretical Artificial Intelligence*, 17(3):161–200, 2005.
- [VCP⁺95] Manuela Veloso, Jaime Carbonell, Alicia Perez, Daniel Borrajo, Eugene Fink, and Jim Blythe. Integrating planning and learning: The PRODIGY architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7(1):81–120, 1995.

31 RePOP

WEBSITE

Authors

Links REPOP WEBSITE REPOSITORY LINK

RelatedTo

Summary Summary description (400 words) goes here.

Description Detailed description goes here.

31.1 RePOP 2001

Description Best Described in [NK64].

References

- [NK64] X. Nguyen and S Kambhampati. Reviving partial order planning. In B. Nebel, editor, *IJCAI-01*, Seattle, WA., 459-464. Morgan Kaufmann Publishers. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.

32 System R

Authors		
Links	WEBSITE UNAVAILABLE	REPOSITORY LINK
RelatedTo		
Summary	Based upon the STRIPS planner. Forward heuristic search. 1) Allows backtracking (not allowed in the STRIPS planner?). 2) Eliminates cycles in sub-goals. 3) Enhanced sub-goal selection. 4) Can use hand-coded control rules. 1) STRIPS Summary description (400 words) goes here. [Lin01]	
Description	Detailed description goes here.	

32.1 System R 1.1 (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown].
Setup	<pre> planner.name = R planner.version = 1.1 planner.dirname = r-1_1 planner.fullname = r-1_1 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = r.sh {%DOMAIN_PATH%} {%PROBLEM_PATH%} success.test.command = grep -q 'Solution:' {%OUTPUT_PATH%} </pre>
Notes	<p>Driver file: "r.sh"</p> <p>(02/15/2006) MCR So apparently, the 64 bit machines I use have different shared library setup that is slightly different from the rest of the world (aka my machine). So, I had to modify the link to the less desirable: r-1_1/pl-3.2.9/lib/libreadline.so.4 -> /usr/lib/libreadline.so.5</p> <p>(02/15/2006) MCR The Prolog system was failing to load the shared library libreadline.so.4. Most likely this is because the authors of the SWI Prolog 3.2.9 directly linked to the shared library name 'libreadline.so.4' instead of to the common name of 'libreadline.so'. In order to get around this issue without requiring a recompile of SWI Prolog 3.2.9, I created the link: r-1_1/pl-3.2.9/lib/libreadline.so.4 -> /usr/lib/libreadline.so I then modified the R driver file, r.sh, to add this path to the LD_LIBRARY_PATH. This is a complete hack! But it is better than recompiling SWI Prolog 3.2.9, imho.</p> <p>(07/10/2003) Daylin The R driver file is written to run only if the working directory is the R installation directory. This allows the system to be moved to a different directory without requiring recompilation.</p> <p>(07/10/2003) Daylin I made several changes to the source file "strips.pl". The functionality affected is: 1) output format 2) allowing the specification of a filepath for temporary output (temp files must be unique to allow for concurrent execution).</p> <p>Changes can be found by searching in 'strips.pl' for "Clayton Daylin".</p> <p>(07/08/2003) Daylin R is dependent upon SWI Prolog 3.2.9, and 3.2.9 source is not fully compatible with later versions of SWI Prolog. Anticipating that this old version of Prolog will not be used for other systems, I have installed it under the R installation directory, in "r-1_1/pl-3.2.9".</p>
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

References

[Lin01] Fangzhen Lin. A planner called R. *Artificial Intelligence*, 22(3):73–76, Fall 2001.

33 SAPA

Authors	Minh B. Do and Subbarao Kambhampati
Links	SAPA WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here. PDDL 2.1
Description	Detailed description goes here.
References	Sapa: A Scalable Multi-Objective Metric Temporal Planner Minh B. Do and Subbarao Kambhampati. JAIR 2003. Improving the Temporal Flexibility of Position Constrained Metric Temporal Plans. Minh B. Do and Subbarao Kambhampati. ICAPS 2003. Sapa: A Domain-Independent Heuristic Metric Temporal Planner Minh B. Do and Subbarao Kambhampati. ECP 2001. Planning Graph-based Heuristics for Cost-sensitive Temporal Planning. Minh B. Do and Subbarao Kambhampati. AIPS 2002. Improving the Temporal Flexibility of Position Constrained Metric Temporal Planning. Minh B. Do and Subbarao Kambhampati. Temporal Planning Workshop, AIPS 2002. Sapa: A Multi-objective Heuristic Metric Temporal Planner. Minh B. Do, Subbarao Kambhampati, and Dan Bryce. Technical Report.

33.1 SAPA 2 (200406) (BUS-4.0 Enabled)

Description Best Described in [DK03].

References

[DK03] Minh Binh Do and Subbarao Kambhampati. Sapa: A multi-objective metric temporal planner. *JAIR*, 20:155–194, 2003.

34 SATPLAN

Authors	Henry Kautz, Bart Selman
Links	SATPLAN (1992-1997) WEBSITE REPOSITORY LINK BLACKBOX (1997-2003) WEBSITE REPOSITORY LINK SATPLAN (2003-2006) WEBSITE REPOSITORY LINK
RelatedTo	
1992-1996 Summary	Summary description (400 words) goes here.
1997-2003 Summary	“BlackBox converts planning problems into Boolean satisfiability problems, which are then solved using a variety of different techniques. The user indicates which techniques should be tried in what order. In constructing the satisfiability problem, blackbox uses the planning graph constructed as in Graphplan. ” [HD02]
1997-2003 Summary	
Description	Detailed description goes here. [?]
Reference	Detailed references taken from version 3.6b README file R. J. Bayardo Jr. and R. C. Schrag (1997). Using CSP look-back techniques to solve real world SAT instances. Proc. AAAI-97. A. Blum and M.L. Furst (1995). Fast planning through planning graph analysis. Proc. IJCAI-95. M.D. Ernst, T.D. Millstein, and D.S. Weld (1997). Automatic SAT-compilation of planning problems. Proc. IJCAI-97. Carla P. Gomes and Bart Selman (1997). Problem Structure in the Presence of Perturbations. Proc. AAAI-97. Carla P. Gomes, Bart Selman, and Henry Kautz (1998). Boosting Combinatorial Search Through Randomization. Proc. AAAI-98. Henry Kautz and Bart Selman (1992). Planning as Satisfiability. Proc. ECAI-92. Kautz, H. and Selman, B. (1996). Pushing the Envelope: Planning, Propositional Logic, and Stochastic Search. Proc. AAAI-96. Henry Kautz and Bart Selman (1998). The Role of Domain-Specific Knowledge in the Planning as Satisfiability Framework. (Figures.) Proc. AIPS-98, Pittsburgh, PA. Henry Kautz and Bart Selman (1998). BLACKBOX: A New Approach to the Application of Theorem Proving to Problem Solving. Working notes of the Workshop on Planning as Combinatorial Search, held in conjunction with AIPS-98, Pittsburgh, PA, 1998. Chu Min Li and Anbulagan (1997). Heuristics based on unit propagation for satisfiability problems. Proc. IJCAI-97. Drew McDermott and the AIPS-98 Planning Competition Committee. PDDL - The Planning Domain Definition Language, Draft 1.6, June 1998. Bart Selman, Hector Levesque and David Mitchell (1992). A New Method for Solving Hard Satisfiability Problems. Proc. AAAI-92. Bart Selman, Henry Kautz, and Bram Cohen (1994). Noise Strategies for Improving Local Search. Proc. AAAI-94.

34.1 SATPLAN 1.0

Description Best Described in [KS92].

34.2 SATPLAN 1.1

Description Best Described in [KS92].

34.3 SATPLAN 1.2

Description	Best Described in [KS92].
-------------	---------------------------

34.4 BlackBox 2.0

Description	Best Described in [KS92].
-------------	---------------------------

34.5 BlackBox 2.5 (BUS-3.0 Enabled) (BUS4.0 Error)

Description	Best Described in [Citation Unknown].
Notes	

34.6 BlackBox 3.6b (BUS3.0 Enabled) (BUS4.0 Error)

Description	Best Described in [Citation Unknown].
Notes	

34.7 BlackBox 4.2 (BUS-4.0 Enabled)

Additional Authors	Henry Kautz, Bart Selman, Shane Neff
Description	The original description is given in [?] and [?].
Setup	<pre> planner.name = BLACKBOX planner.version = 4.2 planner.dirname = blackbox-4.2 planner.fullname = blackbox-4.2 planner.encoding = SAT working.directory = {%system.directory%}/{%planner.dirname%} exec.command = blackbox -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} -noopt success.test.command = grep -q 'total actions in plan' {%OUTPUT_PATH%} </pre>
Notes	<p>20030702 Daylin</p> <p>The following problem appears to have been fixed in Blackbox 4.2.</p> <p>The message "cannot load operator file" at the end of standard output can be caused by the domain filepath being too long (exceeding PATH_MAX?). The error comes from "graphplan.c", and results from the failure of "fopen" for any reason. (Wouldn't it be nice if the cause of the failure were provided in the error message?)</p>
Feature Summary	<pre> plan-ipc = NA plan-year = 2002 plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = CPP plan-tech-gp = true plan-tech-sat = true plan-srch-direction = UNDEF plan-alg-wastar-w = UNKNOWN plan-lang-strips = true plan-rep-prop = true plan-rep-sat = true plan-sol-para = true </pre>

34.8 BlackBox 4.3 (BUS-4.0 Error)

Additional Authors	Henry Kautz, Bart Selman, Shane Neff
Description	The original description is given in [?] and [?].
Setup	<pre> planner.name = BLACKBOX planner.version = 4.2 planner.dirname = blackbox-4.2 planner.fullname = blackbox-4.2 planner.encoding = SAT working.directory = {%system.directory%}/{%planner.dirname%} exec.command = blackbox -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} -noopt success.test.command = grep -q 'total actions in plan' {%OUTPUT_PATH%} </pre>
Notes	<p>20030702 Daylin</p> <p>The following problem appears to have been fixed in Blackbox 4.2.</p> <p>The message "cannot load operator file" at the end of standard output can be caused by the domain filepath being too long (exceeding PATH_MAX?). The error comes from "graphplan.c", and results from the failure of "fopen" for any reason. (Wouldn't it be nice if the cause of the failure were provided in the error message?)</p>
Feature Summary	<pre> plan-ipc = NA plan-year = 2002 plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = CPP plan-tech-gp = true plan-tech-sat = true plan-srch-direction = UNDEF plan-alg-wastar-w = UNKNOWN plan-lang-strips = true plan-rep-prop = true plan-rep-sat = true plan-sol-para = true </pre>

34.9 SATPLAN 2004 (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown].
Setup	<pre> planner.name = SATPLAN planner.version = 2004. planner.dirname = satplan-2004 planner.fullname = satplan-2004 planner.encoding = SAT working.directory = {%system.directory%}/{%planner.dirname%}/Satplan_Linux_Binaries/ exec.command = satplan -domain {%DOMAIN_PATH%} -problem {%PROBLEM_PATH%} exec.cleanup = rm *.soln success.test.command = grep -q 'Solved' {%OUTPUT_PATH%} </pre>
Notes	
Feature Summary	<pre> plan-ipc = IPC4 plan-year = 2004 plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = CPP plan-tech-gp = true plan-tech-sat = true plan-srch-direction = UNDEF plan-alg-wastar-w = UNKNOWN plan-pre-clausered = true plan-pre-encact = true plan-lang-strips = true plan-rep-prop = true plan-rep-sat = true plan-sol-para = true </pre>

34.10 SATPLAN 2006 (BUS-4.0 Enabled)

Description	Best Described in [KSH06]. Modified SatPlan-2004 to allow both action-based encoding and fluent encoding into propositional formulae. Also extended the pruning techniques
Setup	<pre> planner.name = SATPLAN planner.version = 2006. planner.dirname = satplan-2006 planner.fullname = satplan-2006 planner.encoding = SAT working.directory = {%system.directory%}/{%planner.dirname%}/SatPlan2006_LinuxBin/ exec.command = satplan -domain {%DOMAIN_PATH%} -problem {%PROBLEM_PATH%} exec.cleanup = rm *.soln success.test.command = grep -q 'Solved' {%OUTPUT_PATH%} </pre>
Notes	
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

References

- [HD02] Adele Howe and Eric Dahlman. A critical assessment of benchmark comparison in planning. *JAIR*, 17:1–33, July 2002.
- [KS92] Henry Kautz and Bart Selman. Planning as satisfiability. In *Proceedings ECAI-92*, 1992.
- [KSH06] Henry Kautz, Bart Selman, and Joerg Hoffmann. Satplan: Planning as satisfiability. In *Abstracts of the 5th International Planning Competition, ICAPS-06.*, 2006.
-

35 SGP - Sensory Graphplan

Authors	Daniel S. Weld, Corin R. Anderson, and David E. Smith
Links	SGP WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Graphplan with Sensing and Uncertainty. Produces Contingent Plans. Summary description (400 words) goes here.
Capabilities	1) STRIPS 2) Universal quantification 3) Existential quantification 3) Conditional effects 4) Negated preconditions 5) Sensing actions
Related Planners	1) SGP is an extension of CGP [Wel99] 2) SGP implements a superset of UCPOP functionality, and is faster [WAS98].
Description	Detailed description goes here.

35.1 SGP 1.0b (BUS3.0) (BUS-4.0 Enabled)

Description	This is the September 5, 1998 version; it is most likely the closest to the version used for the competition, though it is impossible to tell by the current writing. Best Described in [Citation Unknown].
Setup	<pre> planner.name = SGP planner.version = 1.0b planner.dirname = sgp-1.0b planner.fullname = sgp-1.0b planner.encoding = GRPL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = {%lisp%} -L "loader" -e '(progn (load-gp) (load "driver") (sgp :domain-filepath "%DOMAIN_PATH%"))' success.test.command = grep 'space allocation' {%OUTPUT_PATH%} -B5 head -n1 grep -q -v NIL </pre>
Notes	<p>(01/08/2007) MCR Copied the SGP_1.0h settings and recompiled the planner for ACL 8.0.</p> <p>(06/26/2003) Daylin</p> <p>Driver file: "driver.lisp"</p> <p>Compiling:</p> <ol style="list-style-type: none"> 1) delete *.fasl in "sgp-1.0h" directory 2) update *gp-dir* in "loader.lisp" 3) (load "loader.lisp") 4) (compile-gp) 5) (compile-file "loader.lisp") 6) (compile-file "driver.lisp") <p>SGP is compiled to run only if the working directory is the SGP installation directory. This allows the system to be moved to a different directory without requiring recompilation.</p>
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

35.2 SGP 1.0h (BUS-4.0 Enabled)

Description	This is the January 14, 2000 version. Best Described in [Citation Unknown].
Setup	<pre>planner.name = SGP planner.version = 1.0h planner.dirname = sgp-1.0h planner.fullname = sgp-1.0h planner.encoding = GRPL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = {%lisp%} -L "loader" -e '(progn (load-gp) (load "driver") (sgp :domain-filepath "{%DOMAIN_PATH%}" success.test.command = grep 'space allocation' {%OUTPUT_PATH%} -B5 head -n1 grep -q -v NIL</pre>
Notes	<p>(06/26/2003) Daylin</p> <p>Driver file: "driver.lisp"</p> <p>Compiling:</p> <ol style="list-style-type: none">1) delete *.fasl in "sgp-1.0h" directory2) update *gp-dir* in "loader.lisp"3) (load "loader.lisp")4) (compile-gp)5) (compile-file "loader.lisp")6) (compile-file "driver.lisp") <p>SGP is compiled to run only if the working directory is the SGP installation directory. This allows the system to be moved to a different directory without requiring recompilation.</p>
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

- [WAS98] Daniel S. Weld, Corin R. Anderson, and David E. Smith. Extending Graphplan to handle uncertainty and sensing actions. In *Proc. AAAI-98*, pages 897–904, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.
- [Wel99] Daniel S. Weld. Recent advances in AI planning. *AI Magazine*, 20(2):93–123, 1999.
-

36 SGPlan

Authors	Yixin Chen, Chih-Wei Hsu, and Benjamin W. Wah
Links	SGPLAN WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here. PDDL 2.2
References	Y. Chen and B. W. Wah, Automated Planning and Scheduling using Calculus of Variations in Discrete Space, Proc. of Intl Conf. on Automated Planning and Scheduling, 2003, pp. 2-11. J. Koehler, J. Hoffmann, On Reasonable and Forced Goal Orderings and their Use in an Agenda-Driven Planning Algorithm, Journal of Artificial Intelligence Research, Volume 12, 2000, pp. 338-386. J. Porteous, L. Sebastia, J. Hoffmann, On the Extraction, Ordering, and Usage of Landmarks in Planning, Proc. 6th European Conf. on Planning, 2001, pp. 37-48. B. W. Wah and Y. Chen, Partitioning of Temporal Planning Problems in Mixed Space using the Theory of Extended Saddle Points, Proc. 15th IEEE Intl Conf. on Tools with Artificial Intelligence, 2003, pp. 266-273.

36.1 SGPlan 4.0 (June 2004) (BUS-4.0 Enabled)

Description	Best Described in [?]. This appears to be the version entered into IPC4. The website [Citation Unknown] lists the date as June 2004.
Setup	<pre> planner.name = SGPlan planner.version = 2004 planner.dirname = sgplan-2004 planner.fullname = sgplan-2004 planner.encoding = Hybrid working.directory = {%system.directory%}/{%planner.dirname%} exec.command = sgplan -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} exec.cleanup = rm *.soln success.test.command = grep -q 'Solution found' {%OUTPUT_PATH%} </pre>
Notes	<p>2005-10-04 SGPlan appears to need more than 256M. It gets killed immediately with this memory limit. A sample run that takes more than a few seconds seems to take about 500M of memory. On the positive side, the planner seems to solve a large number of problems when given appropriate memory.</p>
Feature Summary	<pre> plan-ipc = IPC4 plan-year = 2004 plan-age = 1 plan-programmers = 15 plan-version = 1 plan-impl-lang = C plan-tech-rgp = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-pre-subgoal = true plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-typing = true plan-lang-equality = true plan-lang-dis-pre = true plan-lang-cond-eff = true plan-lang-fluents = true plan-lang-pddl22 = true plan-lang-til = true plan-rep-prop = true plan-rep-rgp = true plan-rep-clsdwrlld = true plan-agda-landmark = true </pre>

36.2 SGPlan 4.1 (August 2006) (BUS-4.0 Enabled)

Description	Best Described in [?]. This appears to be the version entered into IPC5.
Setup	<pre>planner.name = SGPlan planner.version = 2006 planner.dirname = sgplan-2006 planner.fullname = sgplan-2006 planner.encoding = Hybrid working.directory = {%system.directory%}/{%planner.dirname%}/sgplan2/ exec.command = sgplan -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} exec.cleanup = rm *.soln success.test.command = grep -q 'Solution found' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

37 SimPlanner

Authors		
Links	SIMPLANNER WEBSITE	REPOSITORY LINK
RelatedTo		
Summary		
Description	Detailed description goes here.	

37.1 SimPlanner 1.0

Description	Best Described in [?]. “This first version of simplanner allows to simulate the execution of a plan, to introduce unexpected events and to repair the plan when it becomes invalid. The efforts were focused on the replanning algorithm.” [Sap06]
-------------	--

37.2 SimPlanner 2.0 (BUS-4.0 Enabled)

Description	Best Described in [?]. “This second version has its own planner. The planning algorithm was developed to be able to work interleaved with the plan execution (see paper). However, this tool has been used only as a classical planner. This is the version wich participated in the International Planning Competition 2002 (the competition results and a brief abstract about this version of SimPlanner can be found here).” [Sap06]
Setup	<pre>planner.name = SimPlanner planner.version = 2.0 planner.dirname = simplanner-2.0 planner.fullname = simplanner-2.0 planner.encoding = RGP working.directory = {%system.directory%}/{%planner.dirname%} exec.command = simplanner -o {%DOMAIN_PATH%} -f {%PROBLEM_PATH%} success.test.command = grep -q 'FINAL PLAN' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

37.3 SimPlanner 3.0 (BUS-4.0 Unchecked)

Description	<p>Best Described in [?]. “The third version is been developed in order to improve the previous version for working in more realistic environments (see paper). The main improvements of this new approach are:</p> <ol style="list-style-type: none">1. Improvements in the planning algorithm in order to obtain plans with more quality.2. Use of an anytime approach: allows to work in time-limited situations.3. Support for numerical functions, sensory actions and uncertainty.4. Inclusion of a graphical execution simulation.” [Sap06]
-------------	---

References

[Sap06] Oscar Sapena. Simplanner publications. Web Page, July 3 2006. Last Accessed January 3, 2007.

38 SNLP

Authors	
Links	SNLP WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

38.1 SNLP 1.0 (BUS-4.0 Enabled)

Description	Best Described in [?].
Setup	<pre>planner.name = SNLP planner.version = 1.0 planner.dirname = snlp-1.0 planner.fullname = snlp-1.0 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%}/ exec.command = {%lisp%} -L "driver" -e '(progn (in-package pddl-to-snlp) (solve "{%DOMAIN_PATH%}" "{%PROBLEM_PATH%}")) success.test.command = grep -q 'Complete!' {%OUTPUT_PATH%}</pre>
Notes	
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

References

39 STAN

Authors	
Links	STAN WEBSITE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

39.1 STAN 1.0 (BUS 4.0 Error)

Description	Best Described in [Citation Unknown]. “The original AIPS’98 version of STAN. Graphplan-based with initial exploitation of types from TIM. Includes wavefront and compact encoding of the graph.” [LF]
Notes	

39.2 STAN 2.3

Description	Best Described in [Citation Unknown]. “The version built after the competition to clean up bugs and prepare STAN for public release.” [LF]
Notes	

39.3 STAN 3.0 (BUS4.0 Enabled)

Description	Best Described in [Citation Unknown]. “This adds Kambhampati’s EBL/DDB and symmetry. There are also bug-fixes and several implementation improvements for greater efficiency.” [LF]
Setup	<pre>planner.name = STAN planner.version = 3 planner.dirname = stan-3 planner.fullname = stan-3 planner.encoding = GRPL working.directory = {%system.directory%}/{%planner.dirname%}/PUBLICSTAN/ exec.command = stan {%DOMAIN_PATH%} {%PROBLEM_PATH%} success.test.command = grep -q 'Time: 1' {%OUTPUT_PATH%}</pre>
Notes	20070108 MCR Initially, bison (GNU Bison) 2.0 had a wierd problem where it mangled the filename pddl.yacc.tab.c to pddl.tab.cacc. I just renamed the file to the proper name and added '-fpermissive' to the compiler to allow some no longer ISO declarations.
Feature Summary	<pre>plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN</pre>

39.4 STAN 3.0s

Description	Best Described in [Citation Unknown]. This is the same planner as listed in Section 39.4 but with Symmetry turned on.
Notes	

39.5 STAN 4.0 (BUS-4.0 Error)

Description	Best Described in [Citation Unknown]. “This is the AIPS’00 version and it includes a simple forward planning engine in addition to the Graphplan engine. The choice between the systems is automatic. It also uses TIM to carry out action filtering and is able to abstract certain kinds of sub-problems (path-planning and limited resource-handling) automatically. This version is now available, but it should be noted that the release version is essentially identical to the AIPS 2000 competition version and is not extensively tested. We know that there are robustness issues and various bugs.” [LF]
Setup	
Notes	
Feature Summary	

References

- [LF] Derek Long and Maria Fox. Stan: An spg planning system. Web Page. Last accessed January 8, 2007.

40 UCPOP

Authors	
Links	UCPOP WEBSITE REPOSITORY LINK
RelatedTo	Extended by SGP(See Section35)
Summary	“We describe the ucpop partial order planning algorithm which handles a subset of Pednault’s ADL action representation. In particular, ucpop operates with actions that have conditional effects, universally quantified preconditions and effects, and with universally quantified goals. We prove ucpop is both sound and complete for this representation and describe a practical implementation that succeeds on all of Pednault’s and McDermott’s examples, including the infamous Yale Stacking Problem.” [PW92]
Description	<p>The following description is quoted from the UCPOP Website [unk].</p> <p>“Note: UCPOP is an aging system - we recommend Sensory Graphplan (SGP) instead. SGP handles a superset of UCPOP functionality and is much, much faster. Common Lisp source code for the UCPOP partial order planner, version 4.1, is available via anonymous FTP. UCPOP operates with actions that have conditional effects and universally quantified preconditions and effects. It accepts universally quantified goals. In addition, UCPOP 4.1 allows domain axioms and predicates that call Common Lisp code to determine satisfiability. With a conservative search strategy UCPOP is both sound and complete for this representation, but one can add aggressive, domain-dependent search control with convenient declarative rules. Our Common Lisp implementation is simple enough for classroom use, yet quite efficient (requiring between 2-20ms to explore and refine a partial plan).</p> <p>Version 4.1 replaces and subsumes version 2.0. Highlights include:</p> <ol style="list-style-type: none">1. Twice as fast at plan elaboration2. The revamped, improved graphical Plan Debugger (PDB) (requires CLIM 2.0)3. Safety constraints (as explained here).4. Improved quantification over dynamic universes5. More sophisticated search control strategies (including Schubert and Gerevini’s ZLIFO)6. Improved user’s manual <p>This builds on the features introduced in version 2.0:</p> <ol style="list-style-type: none">1. Declarative specification of search control rules2. Universal quantification over dynamic universes (object creation and destruction)3. Domain axioms4. Procedural attachment (predicates expanding to lisp code).5. Larger set of domain theories & search functions for testing. “[unk]

40.1 UCPOP 2.0

Description	Best Described in [Citation Unknown].
-------------	---------------------------------------

40.2 UCPOP 4.1 (BUS-4.0 Enabled)

Description	Best Described in [Citation Unknown].
Setup	<pre>planner.name = UCPOP planner.version = 4.1 planner.dirname = ucpop-4.1 planner.fullname = ucpop-4.1 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = {%lisp%} -L "bus-loader" -e '(progn (load-ucpop) (in-package :ucpop) (ucpop::do-ucpop :domain success.test.command = grep -q 'Complete' {%OUTPUT_PATH%}</pre>
Notes	<p>2006-01-05 MCR</p> <p>This planner was in BUS3fin. The definition (as best we can tell) was:</p> <pre>(define-planner ("UCPOP" "4.1") :time-intercept 259.7735 :time-inits-coefficient -0.5832 :time-objects-coefficient 14.7113 :time-goals-coefficient 0.0 :time-operators-coefficient 64.8468 :time-predicates-coefficient -67.5351 :goals-intercept 0.6351 :goals-slope -0.0624 :inits-intercept 0.7799 :inits-slope -0.0034 :objects-intercept 0.7635 :objects-slope -0.0051 :operators-intercept 0 :operators-slope 0 :predicates-intercept 0.7921 :predicates-slope -0.0185 :probability-divisor 4 :exec-string (concatenate 'string "exec /usr/local/allegro5.0.1/lisp -I " *current-planner-directory* "ucpop.dxl -e '(progn (in-package :ucpop) (ucpop::do-bus))' -kill > /tmp/ucpop.out" :success-test-string "exec /usr/xpg4/bin/grep -q \"Complete\" /tmp/ucpop.out" :dump-file-name "/tmp/ucpop.out")</pre> <p>Driver file: "driver.lisp"</p> <p>(06/26/2003) Daylin</p> <p>Compiling:</p> <ol style="list-style-type: none">1) delete *.fasl in "ucpop-4.1" directory2) update *ucpop-dir* in "bus-loader.lisp" and "loader.lisp"3) (load "bus-loader.lisp")4) (compile-ucpop) <p>UCPOP is compiled to run only if the working directory is the UCPOP installation directory. This allows the system to be moved to a different directory without requiring recompilation.</p>

Feature Summary	<pre> plan-ipc = NA plan-year = 1992 plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = LISP plan-tech-pocl = true plan-srch-direction = AGENDA plan-alg-wastar-w = UNKNOWN plan-lang-strips = true plan-lang-adl = true plan-lang-exis-pre = true plan-lang-univ-pre = true plan-lang-quan-pre = true plan-lang-cond-eff = true plan-rep-pocl = true plan-agda-ucpop = true plan-sol-para = true </pre>
--------------------	---

References

- [PW92] J. S. Penberthy and D. Weld. UCPOP: A sound, complete, partial-order planner for ADL. In *Proceedings 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*, Cambridge, MA, October 1992.
- [unk] The ucpop planner. Web Page.

41 VHPOP

Authors		
Links	VHPOP WEBSITE	REPOSITORY LINK
RelatedTo		
Summary	Summary description (400 words) goes here.	
Description	Detailed description goes here.	

41.1 VHPOP 2.2 (BUS-4.0 Enabled)

Description	Best Described in [YS03].
Setup	<pre> planner.name = VHPOP planner.version = 2.2 planner.dirname = vhpop-2.2 planner.fullname = vhpop-2.2 planner.encoding = POCL working.directory = {%system.directory%}/{%planner.dirname%} exec.command = vhpop {%DOMAIN_PATH%} {%PROBLEM_PATH%} success.test.command = grep -q 'Time:' {%OUTPUT_PATH%} </pre>
Notes	<p>(2003) Daylin</p> <p>The following compilation error is due to a bug in gcc 3.2. The bug is fixed in gcc 3.2.2.</p> <pre> " ...pthread.h:163: parse error before ‘__thread’ " </pre>
Feature Summary	<pre> plan-ipc = NA plan-year = UNKNOWN plan-age = UNKNOWN plan-programmers = UNKNOWN plan-version = UNKNOWN plan-impl-lang = UNKNOWN plan-srch-direction = UNKNOWN plan-alg-wastar-w = UNKNOWN </pre>

References

- [YS03] H.L.S. Younes and R.G. Simmons. VHPOP: Versatile heuristic partial order planner. *JAIR*, 20:405–430, 2003.

42 YAHSP - Yet Another Heuristic Search Planner

Authors	
Links	YAHSP WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

42.1 YAHSP version

Description	Best Described in [Vid04].
-------------	----------------------------

References

- [Vid04] V. Vidal. A lookahead strategy for heuristic search planning. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS-04)*, pages 150–159, Whistler, CA, June 2004.
-

43 Zeno

Authors	
Links	ZENO WEBSITE UNAVAILABLE REPOSITORY LINK
RelatedTo	
Summary	Summary description (400 words) goes here.
Description	Detailed description goes here.

43.1 Zeno VERSION

Description Best Described in [PW94].

References

- [PW94] J.S. Penberthy and D. Weld. Temporal planning with continuous change. In *AAAI-94*, July 1994.

Part IV

DOMAINS AND PROBLEMS

- 44 Summary by Year
- 45 Summary by Domain Name
- 46 PDDL Language Features

47 AIPS1998 - First International Planning Competition (IPC1)

AIPS 1998 WEBSITE REPOSITORY PDDL 1.0 defined in [AIP98].

Description: The descriptions of the domains are taken from the website [McD98]. All problems above have generators written in LISP. Authorship of the code appears to be Drew McDermott with the exception of randogrid.lisp, which McDermott cites was co-authored with Manuela Veloso.

47.1 Movie

Origin TODO

Adaptions None.

Description “In this domain, the goal is always the same (to have lots of snacks in order to watch a movie), but the number of constants increases with problem number. Some planners have combinatorial problems in such cases. This domain was created by Corin Anderson.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

47.2 Gripper

Origin TODO

Adaptions None.

Description “There is a robot with two grippers. It can carry a ball in each. The goal is to take N balls from one room to another; N rises with problem number. Some planners treat the two grippers asymmetrically, giving rise to an unnecessary combinatorial explosion. This domain was created by Jana Koehler.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

47.3 Logistics

Origin TODO

Adaptions None.

Description “There are several cities, each containing several locations, some of which are airports. There are also trucks, which can drive within a single city, and airplanes, which can fly between airports. The goal is to get some packages from various locations to various new locations. This domain was created by Bart Selman and Henry Kautz, based on an earlier domain by Manuela Veloso.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

47.4 Mystery

Origin “This domain was created by Drew McDermott.” [McD98]

Adaptions None.

Description “There is a planar graph of nodes. At each node are vehicles, cargo items, and some amount of fuel. Objects can be loaded onto vehicles (up to their capacity), and the vehicles can move between nodes; but a vehicle can leave a node only if there is a nonzero amount of fuel there, and the amount decreases by one unit. The goal is to get cargo items from various nodes to various new nodes. To disguise the domain, the nodes were called emotions, the cargo items were pains, the vehicles were pleasures, and fuel and capacity numbers were encoded as geographical entities. ” [McD98]

Problem Generator Written in LISP. Original file is “complete-repository/FULL-DISTROS/aipscomp98/g This code was revised and ported to C in the ff domain collection (See Section 52).

Parameters

name

size

edgeprob

fuelprob

fuelamt

neighfac

vehprob

cargopro

cargonum

numgoals
clumpprob

Generation

47.5 Mprime

Origin TODO

Adaptions None.

Description “This is the mystery domain with one extra action, the ability to squirt a unit of fuel from any node to a neighboring node, provided the originating node has at least two units. Created by Drew McDermott.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

47.6 Grid

Origin TODO

Adaptions None.

Description “There is a square grid of locations. A robot can move one grid square at a time horizontally and vertically. If a square is locked, the robot can move to it only by unlocking it, which requires having a key of the same shape as the lock. The goal is to get keys from various locations to various new locations. This domain was created by Jana Koehler, based on an earlier domain by Drew McDermott.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

47.7 Assembly

Origin TODO

Adaptions None.

Description “The goal is to assemble a complex object made out of subassemblies. The sequence of steps must obey a given partial order. In addition, through poor engineering design, many subassemblies must be installed temporarily in one assembly, then removed and given a permanent home in another. This domain was created by Drew McDermott.” [McD98]

Problem Generator TODO

Parameters TODO

Generation TODO

48 AIPS2000 - Second International Planning Competition (IPC2)

WEBSITE

Description The description of the five domains for this competition are taken directly from [?]. Bacchus writes “There were 5 problem domains and on the order of 50 to 100 test problems in each domain. The test suite contained a sequence of problems of increasing size and complexity. The first two domains were standard domains, designed to help the competitors become accustomed with the mechanics of the competition.” There were no problem generators included with these data files.

48.1 Blocks-world

Origin TODO

Adaptions None.

Description “The first domain was the standard fouroperator blocks-world domain (all STRIPS operators). The blocks world is probably the oldest AI planning domain and has come under increasing criticism as a prototypically uninteresting and unrealistic toy domain, something AI should move beyond. However, in experiments I conducted with the planners in the 1998 competition, and older planning systems, I found that none of them were able to solve blocks-world problems effectively. Typically, they hit a computational cliff once they got to problems containing 13 to 14 blocks. However, the blocks world has a simple and natural structure that intelligent systems should be able to take advantage of, just as humans can. I have no objection to discarding a toy problem that contains no natural structure or is easy to solve, but it seems to be scientifically unsatisfactory to discard a toy problem simply because it is embarrassing to admit that it can’t be solved with current techniques. This said, I am pleased to report that now the field can in good conscience leave the blocks world behind; as the reader can see, many of the planners in the 2000 competition performed well on this domain.” [?]

Problem Generator TODO

Parameters TODO

Generation TODO

48.2 Logistics

Origin TODO

Adaptions None.

Description “The second domain was the logistics world, a domain that was used in the 1998 competition. This domain encodes a very simplified form of logistics planning. In particular, there are a collection of packages at various locations in various cities. The packages can be transported between locations in the same city by trucks and between airports in different cities by airplanes. The goal is to move the various packages from their initial location to a final location. The domain contains six STRIPS operators, and its main complexity comes from the size of the state space. On some of the larger problems, the search space can have branching factors as large as 1000 and required plans containing over 200 steps. This domain proved to be hard for the systems in the 1998 competition but much easier for the systems in the 2000 competition. The improvement in performance over the two-year period is remarkable.” [?]

Problem Generator TODO

Parameters TODO

Generation TODO

48.3 Schedule

Origin TODO

Adaptions None.

Description “The third domain was the schedule world, a simplified machining domain. The domain consists of nine machining operators that do things such as paint, grind, and drill an object. The goal is to transform some collection of objects by performing various machining operations on them. Some operations destroy the effects or preconditions of other operations; for example, grinding destroys the paint on an object, and rolling an object makes it hot, after which there is no way to polish it. Furthermore, the domain has a simple notion of time: An object cannot be operated on by two different machines at the same time, and a machine cannot operate on two different objects at the same time. A correct plan has to sequence these operations in the right way and account for the resource constraints. Most of the operators in this domain are ADL operators with conditional and universal effects.” [?]

Problem Generator TODO

Parameters TODO

Generation TODO

48.4 FreeCell

Origin TODO

Adaptions None.

Description “The fourth domain was an encoding of the solitaire card game FreeCell. This free computer game comes with every WINDOWS system operating it has been claimed to be the most popular computer game. The game layout is shown in figure 1. The four squares at the upper right are the home cells, one for each suit. The goal is to move all the cards home in the order ace, , King. The four squares at the upper left are the free cells. They act as buffers, a temporary space to hold as many as four different cards. The main board contains eight columns of cards, randomly configured. In this section of the board, only exposed cards can be moved. They can be moved to a home cell if all previous cards are already there (for example, in figure 1 if any of A A, Awo, , or 2 were exposed, they could be moved home); moved to a free cell if there is space available; or moved in a legal solitaire move, that is, moved on top of a next-highest card of opposite color (for example, in figure 2, the 3 could be moved on top of the 4, thus exposing the 10). This domain was encoded using 10 STRIPS operators. Starting with a fixed set of initial games, simpler versions of each game were generated by removing the top k cards of every suit. For example, by removing the face cards in this configuration, we create a simpler game containing only 40 cards. The complexity of the game reduces considerably as the number of cards is decreased: Fewer deadlocks are created. It can be proved that this domain contains unsolvable games, although they are relatively rare.” [?]

Problem Generator TODO

Parameters TODO

Generation TODO

48.5 Miconic

Origin TODO

Adaptions None.

Description “The fifth and final domain was an encoding of a sophisticated elevator controller produced by Schindler Lifts Ltd. This controller, in operation in some of the s tallest buildings, controls a bank of elevators. Users go to a central station and punch in their floor request. The controller then tells them which elevator they should use. In this way, the controller attempts to minimize the delay time by scheduling an entire bank of elevators together instead of having each elevator run separately. The controller also allows for various special situations, for example, VIP users or sets of users who should not be riding an elevator together. The planning domain was configured in a number of different ways to accommodate the representational power of different planning systems.” [?]

Problem Generator TODO

Parameters TODO

Generation TODO

49 ICAPS2002 - Third International Planning Competition (IPC3)

WEBSITE

The following description is from the website [LF03]. Fox and Long write:

The competition was run with a series of domains, divided into three broad collections: transportation related domains, space-applications related domains and a collection of odds-and-ends. The first collection contained the domains called Depots, DriverLog and ZenoTravel, the second contained Satellite and Rovers and the third contained FreeCell, Settlers and UMTranslog-2. Apart from the third set, in each case a domain was represented by several variants, including STRIPS, numeric, simple-timed, timed and, in some cases, more complex problems (usually combining time and numbers in more interesting ways).

Simple-timed problems used durative actions in which durations were fixed for all instances of an action schema, being given as domain constants. Timed versions were more complex, since the durations could depend on problem-specific data. Only in more complex versions could the values on which these durations depended be fluents, since this requires of a planner the capability for managing both durative actions and numeric-valued fluents.

Variants of the domains changed the nature of the problems very significantly, so each domain variant can be seen as a separate domain in its own right. The following sections detail the domains we used in all their variations. [LF03].

49.1 Depots

Origin TODO

Adaptions None.

Description “This domain was devised in order to see what would happen if two previously well-researched domains were joined together. These were the logistics and blocks domains. They are combined to form a domain in which trucks can transport crates around and then the crates must be stacked onto pallets at their destinations. The stacking is achieved using hoists, so the stacking problem is like a blocks-world problem with hands. Trucks can behave like ‘tables’, since the pallets on which crates are stacked are limited.

Strips: This is the basic combined domain.

Numeric: Trucks are equipped with load capacities and consume fuel as they move. Crates have weights and hoists consume fuel as they lift packages. The task is always to achieve the goals at least fuel cost.

Simple-time: durations vary from 10 for driving to 1 for dropping a crate onto a stack. There is considerable scope for concurrency in the use of trucks and hoists.

Time: Durations now depend on distances between locations and the speeds of different trucks. Hoists take different times to load or unload according to their power and the weights of the crates. This makes the concurrency issue more subtle, since faster trucks might be available, but somewhere other than

the location they are needed, so a better plan might involve coordinating concurrent positioning of trucks and use of hoists.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.2 DriverLog

Origin TODO

Adaptions None.

Description “This domain involves driving trucks around delivering packages between locations. The complication is that the trucks require drivers who must walk between trucks in order to drive them. The paths for walking and the roads for driving form different maps on the locations.

Strips: The basic version.

Numeric: Walking and driving are each allocated a cost proportional to the distance travelled by a moving object. Plans must minimize some linear combination of these values. The linear combination is problem instance specific, making it harder for hand-coders to predetermine a strategy for optimising whether drivers walk between trucks or simply drive everywhere.

Hard-Numeric: In this case the fuel consumption of a trucks is proportional to the square of its load. This makes the problem of efficiently using trucks much harder, since it might be better to make multiple trips with smaller loads than to make a single trip with a heavy load.

Simple time: Action durations vary from 20 for walking to 1 for simple boarding and disembarking. Concurrency is possible by using drivers and trucks in parallel, including both movement and loading or unloading activity.

Time: Distances between locations are now used to compute the times taken to travel between locations. This makes the management of concurrent activity rather trickier.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.3 ZenoTravel

Origin TODO

Adaptions None.

Description “The final transportation domain involves transporting people around in planes, using different modes of movement: fast and slow. The key to this domain is that, where the expressive power of the numeric tracks is used, the fast movement consumes fuel faster than slow movement, making the search for a good quality plan (one using less fuel) much harder.

Strips: This version was not particularly challenging, since the domain failed to offer any advantage for using the faster flight method, but still extracted a penalty. The main opportunity for interesting behaviour in this case was to automatically avoid using the inefficient operator altogether. Otherwise, this was a typical transportation problem.

Numeric: In this version aircraft consume fuel at different rates according to the mode of travel. One mode is restricted to planes that are carrying fewer than a threshold number of people, dependent on the plane concerned. Each plane has a separately generated fuel consumption rate in each travel mode.

Simple-time: Using a small set of symbolic fuel levels, this version manages to combine the benefits of fast travel (shorter journey times) with the associated costs (higher fuel consumption) that must be balanced with the cost of refuelling to arrive at time-efficient plans.

Time: In this form the domain uses numbers to encode fuel consumptions, dependent on distances, and speeds to calculate travel times in each travel mode. This version is essentially the domain used to illustrate the Zeno planning system developed by Penberthy and Weld.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.4 Satellite

Origin TODO

Adaptions None.

Description “The first of the domains inspired by space-applications is a first step towards the *Ambitious Spacecraft* described by David Smith at AIPS’00. It involves planning and scheduling a collection of observation tasks between multiple satellites, each equipped in slightly different ways.

Strips: In its Strips version this domain is reasonably straightforward, although it still represents a new benchmark domain.

Numeric: The numeric version introduces fuel costs for satellites to slew between targets and the satellites have a finite (and non-replenishable) fuel source. In addition, data takes up space in a finite capacity data store. The over all effect

of this is to create a constrained bin-packing problem (the constraints being determined partly by the equipment on the different satellites and partly by the fuel limits; the bins being the data stores and the objects to be packed being the observations). Plans are expected to acquire all the necessary data at minimum fuel cost.

Simple-time: This version allows for concurrent use of different satellites, so that the data can be acquired more quickly if they are used efficiently.

Time: Concurrency is complicated by the introduction of different slew times between targets and of calibration times for different instruments paired with different possible calibration targets.

Complex: The Satellite domain offers a complex version, in which the timed version (with varying slew times and calibration times) is combined with the finite data capacities of the numeric-version satellites. The result is a constrained bin-packing problem with the additional requirement that observations be efficiently scheduled (makespan is the quality metric).

Hard Numeric: A final variant used for the Satellite domain is an interesting exploitation of the opportunity to specify a plan quality metric in PDDL2.1. All these problems have no logical goals at all, but the metric judges plans that acquire more data to be better. Therefore, the obvious null plan is of no value, even though it satisfies the (null) goals.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.5 Rovers

Origin TODO

Adaptions None.

Description “Inspired by planetary rovers problems, this domain requires that a collection of rovers navigate a planet surface, finding samples and communicating them back to a lander.

Strips: The Strips version is an interesting challenge in its own right. In addition, a minor subtlety in the encoding is used to prevent parallel communication between rovers and the lander: communication actions precondition on the channel to the lander being free, and then both add and delete this fact. Because deletes occur before adds this has the over all effect of leaving the channel free, but it makes the fact a “moving target” which prevents a concurrent action from using the fact. Some planners find this mechanism difficult to handle.

Numeric: In the numeric version, rovers consume energy in their various activities. They can recharge, but only at locations that are in the sun (they use solar

rechargers). This makes the challenge to manage energy efficiently. Plan quality is judged by the number of recharges required, rather than the plan length.

Simple-time: Concurrent use of the rovers must be coordinated with the bottleneck of communication with the lander over the single communication channel. This makes it a challenge to find efficient makespans.

Time: This variant combined the demands of the simple time durations of activities with the complexities of managing energy levels as in the numeric version. Recharging time depends on the level of charge to be replenished. Plan quality is measured by makespan, but since this reflects the amount of time spent recharging, efficient energy use is also important.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.6 FreeCell

Origin TODO

Adaptions None.

Description “In order to confirm that STRIPS planning is still advancing, we chose this domain from IPC2. It is the familiar solitaire game found on many computers, involving moving cards from an initial tableau, constrained by tight restrictions, to achieve a final suit-sorted collection of stacks.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.7 Settlers

Origin TODO

Adaptions None.

Description “This one was for the numeric track and proved to be a very tough resource management domain. Several interesting issues in encoding arise as well as the subsequent problem of planning with the domain. In particular, resources can be combined to construct vehicles of various kinds. Since these vehicles are not available initially, this is an example of a problem in which new objects are created. PDDL does not conveniently support this concept at present, so it is necessary to name *potential* vehicles at the outset, which can be realised through construction. A very high degree of redundant symmetry exists between these *potential* vehicles, since it does not matter which vehicle names are actually used for the vehicles that are realised in a plan. Planners that begin by grounding all actions can be swamped by the large numbers of potential actions involving these potential vehicles, which could be realised as one of several different types of actual vehicles.

Plan quality is judged by a linear combination of labour use, pollution creation and resource consumption. There is scope for constructing very hard metrics that involve maximising housing construction subject to an increasing pollution penalty (say), to ensure that optimal plan quality is bounded.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.8 UMTranslog-2

Origin TODO

Adaptions None.

Description “This is a numeric domain, but was only used for the hand-coding planners in the competition. It is a large domain with 23 types of objects, 38 predicates, 24 functions and 38 action schemas. Plan quality was measured by plan length (both allowing for concurrent activity and also simply counting steps). More complex metrics would be possible. Thanks go to Dan Wu for her great efforts in encoding this domain and writing the automatic problem generator for it.” [LF03]

Problem Generator TODO

Parameters TODO

Generation TODO

49.9 Additional (non-competition) Domains

“The following domains contain additional examples of PDDL2.1. One is a PDDL version of the old Zenon flight domain and the other is a new domain in which a plan must be found to get a vehicle across a desert. If you find any bugs, have any ideas for new domains, have comments on these domains or have a planner that can solve them, please let us know.

Zeno travel: domain and problems.

Desert Rats: domain and problems. The original problems here are actually rather simpler than was intended - try setting the goal distance to be 600 instead of 300! You might find it helps to add an extra fuel tank, too. In the newer versions of the first problem there is a 500 and a 600 mile target distance to cross. The archive now also contains two other versions of the domain itself - one with a durative refuel and both using conditional effects.

Jugs and Water: domain and a problem. This is the classic problem.

Depots: domain. This archive contains several different versions of a combined logistics and blocks world domain. The versions include numeric and temporal extensions. Now includes a problem generator!

Taxi: domain. This archive contains a numeric domain with durative actions.

DriverLog: Another transportation domain - we promise there will be other things coming! This one is pure STRIPS, so that should keep some of the older planners busy.

These archives contain some sample plans in addition to problems. These will pass through the validator.” [LF03]

50 ICAPS2004 - Fourth International Planning Competition (IPC4)

WEBSITE

Description

Our goals in the creation of the IPC-4 domains were to get as close as possible to applications, and to cover diverse kinds of problem structure. We invested significant effort into achieving these goals, and we believe we succeeded in providing an interesting set of benchmarks.

We contacted various people who developed/are developing planning application domains. In those cases where there was mutual interest in entering the application into IPC-4, we established collaborations, bringing the applications into forms suitable for IPC-4. We took care to select a range of intuitively structurally very different applications. In the process of adapting the applications for use in the IPC-4, inevitably some of the realism had to go, in order to accommodate language or system capabilities. Still, the domains are at least a large step into the right direction.

The authors also pointed out two things:

- 1) All the domains are organized in a two-step hierarchy of domain versions, and domain version formulations. The domain versions differ in terms of the (number of) problem constraints they consider. The domain version formulations differ in terms of the language used to formulate the (same) problem constraints. Formulations of complex problem constraints in more primitive languages such as STRIPS are made by compiling down from the instances formulated in richer languages such as ADL. We let the competitors choose whichever formulation of a domain version they could handle best/handle at all. The results for each domain version were then evaluated together, irrespective of the formulations chosen by the individual planners. Our objective behind this strategy was to enable every planner – even pure STRIPS planners – to tackle a range of interesting problems, while at the same time not introducing too many distinction lines into the competition data.
- 2) Derived predicates are used in PSR to model the flow of electricity, and in Promela to detect deadlock states. Timed initial literals are used in Airport to model the times at which a runway will be blocked due to landing airplanes. They are used in Pipesworld to model goal deadlines. They are used in Satellite to model the times at which data can be sent to earth, and they are used in UMTS to model time windows during which certain set up steps can be accomplished.

IPC-4 featured the following 7 domains; more detailed descriptions of the domains can be found in [HE04]

50.1 Airport

Origin TODO

Adaptions None.

Description “Developed by Jeorg Hoffmann and Sebastian Trug. Planners control the ground traffic on airports. The competition test suites were generated by exporting traffic situations arising during simulation runs in the airport simulation tool Astras (by Wolfgang Hatzack). The largest instances in the test suites are realistic encodings of Munich airport.” [HE03]

Problem Generator

Parameters

Generation

50.2 Pipesworld

Origin TODO

Adaptions None.

Description “Developed by Frederico Liporace and Joerg Hoffmann. Planners control the flow of oil derivatives through a pipeline network, obeying various constraints such as product compatibility, tankage restrictions, and (in the most complex domain version) goal deadlines. One interesting aspect of the domain is that, if one inserts something into the one end of a pipeline segment, something potentially completely different comes out at the other end. This gives rise to several subtle phenomena that can arise in the creation of a plan.” [HE03]

Problem Generator

Parameters

Generation

50.3 Promela

Origin TODO

Adaptions None.

Description “Developed by Stefan Edelkamp. Planners are asked to find deadlocks in communication protocols, translated into PDDL from the Promela specification language. Deadlocks were specified via blocked transitions and processes. The representation chosen for the processes are finite state transition systems, while communication channels are modelled by queues with moving head and tail pointers. The communication protocols used in IPC-4 were the dining philosophers problem, as well as an optical telegraph routing problem.” [HE03]

Problem Generator

Parameters

Generation

50.4 PSR

Origin TODO

Adaptions None.

Description “Developed by Sylvie Thiebaut and Joerg Hoffmann. Planners must resupply a number of lines in a faulty electricity network. The flow of electricity through the network, at any point in time, is given by a transitive closure over the network connections, subject to the states of the switches and electricity supply devices. The domain is therefore a good example of the usefulness of derived predicates in real-world applications.” [HE03]

Problem Generator

Parameters

Generation

50.5 Satellite

Origin TODO

Adaptions None.

Description “Adapted for IPC-4 by Joerg Hoffmann. Planners are asked to collect image data with a number of satellites. The domain was introduced by Maria Fox and Derek Long in IPC-3. All (but one) of the IPC-3 domain versions were re-used with the exact same suites of instances. Two of the IPC-3 domain versions were extended with time windows for the sending of the gathered data to earth, which is the most critical aspect of the problem in reality.” [HE03]

Problem Generator

Parameters

Generation

50.6 Settlers

Origin TODO

Adaptions None.

Description “Introduced by Maria Fox and Derek Long in IPC-3. The domain, i.e. its test suite, could not be solved efficiently by any of the IPC-3 participants. We re-used the domain with no modifications whatsoever, except removing some quantified effects (that could easily be replaced with lists of non-quantified effects).” [HE03]

Problem Generator

Parameters

Generation

50.7 UMTS

Origin TODO

Adaptions None.

Description “Developed by Stefan Edelkamp and Roman Englert. The task is to set up applications for mobile terminals. The objective is to minimize the time needed for the set up, i.e. to minimize the makespan of the plan. When ignoring that objective, i.e. for sub-optimal planners, the problem becomes trivial. For optimal planners, the domain is a realistic challenge.” [HE03]

Problem Generator

Parameters

Generation

51 ICAPS2006 - Fifth International Planning Competition (IPC5)

WEBSITE

Description “Another novelty of the deterministic part of IPC-5 which required considerable efforts concerns the test domains (for more details see the ‘Domains’ page): we designed five new planning domains, together with a large collection of benchmark problems. In order to make PDDL3.0 language more accessible to the competitors, for each test domain, we developed various variants using different fragments of PDDL3.0 with increasing expressiveness. In addition, we re-used two domains from previous competitions, extended with new variants including some of the features of PDDL3.0. The IPC-5 test domains have different motivations. Some of them are inspired by real world applications; others are aimed at exploring the applicability and effectiveness of automated planning for new applications or for problems that have been investigated in other field of computer science; while the domains from previous competitions are used as sample references for measuring the advancement of the current planning systems with respect to the existing benchmarks.” [Ger06]

“The IPC-5 test domains have different motivations. Some of them were inspired by real world applications, (e.g., storage, trucks and pathways); others were aimed at exploring the applicability and effectiveness of automated planning for new applications (pathways), or for known problems that have been addressed in other fields of computer science (TPP and openstacks); finally, two domains were taken from previous competitions, as sample references for the advancement of automated planning with respect to the existing benchmarks (rovers and pipesworld).” [?]

51.1 The Travelling Purchaser

Origin TODO

Adaptions None.

Description “This is a relatively recent planning domain that has been investigated in operations research (OR) for several years, e.g., (Riera-Ledesma & Salazar-Gonzalez 2005). The Travelling Purchaser Problem (TPP) is a known generalization of the Travelling Salesman Problem, and is defined as follows. We have a set of products and a set of markets. Each market can provide a limited amount of each product at a known price. The TPP consists in selecting a subset of markets such that a given demand for each product can be purchased, minimizing the combined travel and purchase cost. This problem arises in several applications, mainly in routing and scheduling contexts, and it is NP-hard. In OR, computing optimal or near optimal solutions for the TPP instances is still an active research topic. For IPC-5, we have formalized several variants of this domain in PDDL. One of them is equivalent to the original TPP, while the others are different formulations or significant (we believe and hope) extensions. In all these domain variants, plan quality is important, although for some instances even finding an arbitrary solution could be quite difficult for a fully-automated planner. For this domain, we developed both a metric version without time and a metric-time version. We begin the description with the metric version because it is the one equivalent to the original formulation of the TPP.

Metric: This version is equivalent to the original formulation of the TPP in OR. There are only three operators, two of which are used to model the purchasing buy-. The first buys at a certain market (?m) the whole amount of a type of goods (?g) sold by the market (?m and ?g are operator parameters); while the second one buys at ?m the amount of ?g that is needed to complete the purchase of ?g (as specified in the problem goals). In this version, every market is directly connected to every other market and to the depots. Moreover, there is only one depot and only one truck.

Propositional This version models a variant of the original TPP where: (1) there can be more than one depot and more than one truck; (2) the amount of goods are discrete and represented by qualitative levels; (3) every type of goods has the same price, independent from the market where we buy it; (4) there are two new operators for loading and unloading goods to/from trucks; (5) markets and depots can be indirectly connected.

Simple Preferences The operators in this domain are the same as in the propositional version. The difference is in the goals, which are all soft goals (preferences). These preferences concern maximizing the level of goods that are stored in the depots, constraints between the levels of different stored goods, and the safety condition that all purchased goods are stored at some market.

Qualitative Preferences The operators in this version are the same as in the propositional version. All goals are preferences concerning maximizing, for every type of goods, the purchased and stored levels. This version includes preferences over trajectory constraints. These are constraints between the levels of two types of stored goods; constraints about the use of the trucks for loading goods; constraints imposing the use of every truck. Moreover, we have the preference that in the final state all purchased goods are stored at some depot.

Metric-Time With respect to the simpler metric version, which is equivalent to the original formulation of the TPP, this version has the following main differences: same as points (1), (4), (5) illustrated in the description of the propositional variants; each action has a duration and the plan quality is a linear combination of totaltime (makespan) and the total cost of traveling and purchasing; the has rate (if you buy the whole amount of a type of goods that is sold at a market, then you have a discount).rebatea buyalloperator

Metric-Time Constraints The operators in this version are the same as in the metric-time version. In addition, in the domain file, we have some strong constraints imposing that in the final state all purchased goods are stored, every market can be visited by at most one truck at the same time, every truck is used. Moreover, in the problem specification, we have several strong constraints about the relative amounts of different types of goods stored in a depot, the number of times a truck can visit a market, the order in which goods should be stored, the order in which we should store some type of goods and buy another one, and deadlines about delivering goods once they have been loaded in a truck.

Complex Preferences The operators in this version are the same as in the metric-time version. In addition, it contains many preferences over state trajectory

constraints that are similar to those used for the metric-time constraints version.” [?]

Problem Generator

Parameters

Generation

51.2 Openstacks

Origin TODO

Adaptions None.

Description “The openstacks domain is based on *minimum maximum simultaneous open stacks* combinatorial optimization problem, which can be stated as follows: A manufacturer has a number of orders, each for a combination of different products, and can only make one product at a time. The total required quantity of each product is made at the same time (because changing from making one product to making another requires a production stop). From the time that the first product included in an order is made to the time that all products included in the order have been made, the order is said to and during this time it requires (a temporary storage space). The problem is to order the making of the different products so that the maximum number of stacks that are in use simultaneously, or equivalently the number of orders that are in simultaneous production, is minimized (because each stack takes up space in the production area).

This problem, and many related variants, have been studied in operations research (see, e.g., Fink & Voss 1999). It is known to be NP-hard, and equivalent to several other problems (Linhares & Yanasse 2002). This is a pure optimization problem: for any instance of the problem, every ordering of the making of products is a solution, which at worst uses as many simultaneously open stacks as there are orders. Thus, finding a plan is quite trivial (in the sense that there exists a domainspecific linear-time algorithm that solves the problem), but finding a plan of high quality is hard (even for a domain-specific algorithm).

The openstacks problem was recently posed as a challenge problem for the constraint programming community, and, as a result, a large library of problem instances, together with results on those instances for a number of different solution approaches, are available (see Smith & Gent (2005)).

Propositional This variant is simply an encoding of the original openstacks problem as a planning problem. The encoding is done in such a way that minimizing the length (sequential or parallel) of the plan also minimizes the objective function, i.e., the maximum number of simultaneously open stacks. There are three basic actions to start orders, make products, and ship orders once they are completed, plus an action a new stack, but in order to ensure the correspondance between parallel length and the objective function, some of these actions are split in two parts. The domain formulation uses some ADL constructs (quantified disjunctive preconditions), but these can be compiled away with only a linear increase in size. The problems are a selection of the

problems used in the constraint modelling challenge, including a few problems that could not be solved (optimally) by any of the CSP approaches, plus a small number of extra small instances.

Time In this variant of the domain the number of available stacks is fixed, and the objective is instead to minimize makespan. Makespan is dominated by the actions that make products. The number of stacks is for each problem chosen to be somewhere between the optimal and the trivial upper bound (equal to the number of orders).

Metric-Time In this variant, the objective function is to minimize a (linear) combination of the number of open stacks and the plan makespan. The number of open stacks is modelled using numeric fluents.

Simple Preferences In this variant, the goal of including all required products in each order is softened, and a “score” (or “reward”) is instead given for each product that is included in an order when it is shipped. The objective is to maximize this score. The maximum number of open stacks is fixed, like in the temporal variant, but at a number slightly less than the optimal number required to satisfy all the requirements of all orders. This version of the domain uses an ADL construct (a quantified conditional effects) that can only be compiled away at an exponential increase in problem size.

Complex Preferences This version, like the previous, has soft goals, but also a variable maximum number of open stacks. The objective is to maximize a linear combination of the score (positive) and the number of open stacks (negative). Also like the previous version, the formulation uses a quantified conditional effect.” [?]

Problem Generator

Parameters

Generation

51.3 Storage

Origin TODO

Adaptions None.

Description “The Storage Domain is a planning domain involving spatial reasoning. Basically, the domain is about moving a certain number of crates from some containers to some depots by hoists. Inside a depot, each hoist can move according to a specified spatial map connecting different areas of the depot. The test problems for this domain involve different numbers of depots, hoists, crates, containers, and depot areas. While in this domain it is important to generate plans of good quality, for many test problems, even finding any solution can be quite hard for domain-independent planners.

Altogether, the different variants of this domain, involve almost all the new features of PDDL3.0. Note that this domain is basically a propositional domain, where the space for

storing crates is represented by PDDL literals. For this domain, instead of a metrictime version, we have time- version (without numerical fluents).

Propositional The domain has five different actions: an action for lifting a crate by a hoist, an action for dropping a crate by a hoist, an action for moving a hoist into a depot, an action for moving a hoist from one area of a depot to another one, and finally an action for moving a hoist outside a depot.

Time This variant is basically the propositional variant where the actions have duration and the plan quality is totaltime (plan makespan).

Simple Preference The operators in this domain are the same as those in the propositional version. The main difference is in the goals. All goals are soft goals (preferences). These preferences concern which depots and depot areas should be used for storing the crates, the desire that crates are stored in the same depot, the desire that the incompatible crates stored in the same depot are located at non-adjacent areas of the depot and, finally, the desire that the hoists are located in depots different from those where we store the crates.

Qualitative Preferences The operators in this domain are the same as those in the propositional version. The differences are in the preferences over the goals and state trajectory constraints. All goals are soft goals similar to some of the soft goals specified in the simple preferences variant. The preferences over trajectory constraints concern constraints about the use of the available hoists for moving the crates, and about the order in which crates are stored in the depots. Moreover, we have the preference that in any state crossed by the plan, the adjacent areas in a depot can be occupied only by compatible crates.

Time Constraints The operators in this version are the same as those in the temporal version. The problem goals are specified by at- constraint imposing that all crates are stored in a depot. The problems have several constraints imposing that a crate can be lifted at most once, ordering constraints about storing certain crates before others, deadlines for storing the crates, and maximum time a hoist can stay outside a depot. There are also constraints imposing a safety condition, that in the final state, all hoists are inside a depot; some constraints imposing that every hoist is used; and some constraints imposing that incompatible crates are not stored at adjacent areas of the depot.

Time Preferences The operators in this version are the same as those in the temporal version. In addition, this version contains many preferences over state trajectory constraints that are similar to those used for the time constraints version.” [?]

Problem Generator

Parameters

Generation

51.4 Trucks

Origin TODO

Adaptions None.

Description “The Trucks Domain Essentially, this is a logistics domain about moving packages between locations by trucks under certain constraints. The loading space of each truck is organized by areas: a package can be (un)loaded onto an area of a truck only if the areas between the area under consideration and the truck door are free. Moreover, some packages must be delivered within a deadline. In this domain, it is important to find good quality plans. However, for many test problems, even finding one plan could be a rather difficult task.

Like the Storage domain, this domain has variant instead of a metric-time variant (i.e., there are no numerical fluents). The other variants make extensive use of the new features of PDDL3.0. We start the description from the time constraint version, because it is the one closest to a realistic problem.

Time Constraints The domain has four different actions: an action for loading a package into a truck, one for unloading a package from a truck, one for moving a truck, and finally one for delivering a package. The durations of loading, unloading and delivering packages are negligible compared to the durations of the driving actions. The problem goals require that certain packages are at their final destinations by certain deadlines. For this variant, we also created an equivalent Time-, in which the trajectory constraints of are compiled into timed initial literals. Each competing team is free to choose one of the two alternative variants.

Time The operators are the same as those in the time constraints version, but there is no deadline for delivering packages. Finding a valid plan in this version is significantly easier, but finding a plan with short makespan is still challenging.

Complex Preferences The operators in this version are the same as those in the constraints version. The deadlines are modeled by preferences. Moreover, this version contains preferences over trajectory constraints. These are constraints imposing some ordering about when delivering packages, constraints about the usage of the areas in the trucks, and constraints about loading packages.

Propositional The operators in this version are similar to those in the constraints version, with the main difference that time is modeled as a discrete resource (with a fixed number of levels). Moreover, the driving actions cannot be executed concurrently.

Simple Preferences The operators in this domain are the same as those in the propositional version. The difference concerns the problem goals where the delivering deadlines are modeled by preferences.

Qualitative Preferences The operators in this domain are the same as those in the propositional version. The difference concerns the problems goals including soft delivering deadlines. Moreover, this version includes many preferences over state trajectory constraints that are similar to those used for the complex preferences version.” [?]

Problem Generator

Parameters

Generation

51.5 Pathways

Origin TODO

Adaptions None.

Description “The Pathways Domain This domain is inspired by the field of molecular biology, specifically biochemical A pathway is a sequence of chemical reactions in a biological organism. Such pathways specify mechanisms that explain how cells carry out their major functions by means of molecules and reactions that produce regular changes. Many diseases can be explained by defects in pathways, and new treatments often involve finding drugs that correct those (Thagard 2003) We can model parts of the functioning of a pathway as a planning problem by simply representing chemical reactions as actions. The goal in these planning problems is to construct a sequence of reactions that produces one or more substances, using a limited number of substances as input. The planner is partly free to choose which input substances to use, i.e., to choose some aspects of the initial state of the problem. This aspect of the problem is modelled by means of additional actions.

The biochemical pathway domain of the competition is based on the pathway of the Mammalian Cell Cycle Control as it described in (Kohn 1999) and modelled in (Chabrier 2003). There are three different kinds of basic actions corresponding to the different kinds of reactions that can appear in a pathway.

Propositional This is a simple qualitative encoding of the reactions of the pathway. The domain has five different actions: an action for choosing the initial substances, an action for increasing the quantity of a chosen substance (in the propositional version, quantity coincides with presence, and it is modeled through a predicate indicating if a substance is available or not), an action modeling biochemical association reactions, an action modeling biochemical association reactions requiring catalysts, and an action modeling biochemical synthesis reactions. Also, there is an additional set actions used to encode the disjunctive problem goals.

The goals refer to substances that must be synthesized by the pathway, and are disjunctive with two disjuncts each. Furthermore, there is a limit on the number of input substances that can be used by the pathway.

Simple Preferences This is similar to the propositional version, with the difference that both the products that must be synthesized by the pathway and the number of the input reactants that are used by the network are turned into preferences. The challenge here is finding plans that achieve a good tradeoff between the different kinds of preferences.

Metric-Time In this version of the domain, reactions have different durations. The reactions can only happen if their input reactants reach some concentration level, and reactions generate their products in specific quantities. The goals in this version are summations of substance concentrations that must be generated by the reactions of the pathway. The plan metric minimizes some linear combination of the number of input substances and the plan duration.

Complex Preferences This is an extension of the metric-time version with different preferences concerning the concentration of substances of the pathway, or the order in which substances are produced. The metric is a combination of these preferences, the number of substances used and the plan makespan.”
[?]

Problem Generator

Parameters

Generation

51.6 Extended Rovers

Origin TODO

Adaptions None.

Description “The Extended Rovers Domain The Rovers domain was introduced in the 2002 planning competition (Long & Fox 2003). It models the problem of planning for a group of planetary rovers to explore the planet they are on (taking pictures and samples from interesting locations).

Propositional and Metric-Time The propositional and metric-time versions of the domain are the same as in IPC 2002, with the addition of some planning problems.

The domain has nine different actions: an action for moving rovers on a planet surface, two actions for sampling soil and rock, an action for dropping rock or soil, an action for calibrating rover instruments, an action for taking image of interesting objective, and finally three actions for transmitting soil data, rock data or image data.

Qualitative Preferences This is the IPC 2002 propositional version with soft trajectory constraints added (constraint types always, sometime and at-most-once are used). The objective is simply to maximize the number of preferences satisfied. The preferences, in the sense that they do not encode preferences on the plan, but are constructed in a way as to make the problem of maximizing the satisfaction of preferences challenging.

Metric Simple Preferences This version is a special case of the complex preferences version, which has preferences only on the goals of the problem.

This version of the domain poses a so-net problem: goals (atoms, and in some cases conjunction of atoms) have values and actions have cost, and the objective is to maximize the sum values of achieved goals minus the sum of costs of actions in the plan. Only the actions that move the rovers have non-zero cost. The domain uses simple (goal state) preferences to encode goal values and fluents to encode action costs. There are three different sets of problems, with somewhat different properties. In the first, goals are interfering, meaning that the cost of achieving any two goals is greater than the sum of achieving

them individually. The second has instead synergy between the goals, i.e., the cost of achieving several goals is less than the sum of achieving each of them separately, while the third contains goals with relationships of both kinds.”
[?]

Problem Generator

Parameters

Generation

51.7 Extended Pipesworld

Origin TODO

Adaptions None.

Description “The Extended Pipesworld Domain The Pipesworld domain was introduced in the previous planning competition (Hoffmann & Edelkamp 2005). It models the transportation of batches of petroleum products in a network of pipelines.

Propositional and Time The propositional and temporal versions of the domain are variant of the domain used in IPC 2004 The domain has six actions: two actions for moving a batch from a tankage to a pipeline segment (one for the start and one for the end of the activity), two actions for moving a batch from a tankage to a pipeline segment, and two actions for moving a batch from a tankage (or pipeline segment) to a pipeline segment (or tankage) in case the pipes consist of only one segment.

Time Constraints The time constraints variant is based on the temporal no-tankage variant from IPC 2004, but adds hard deadlines on when each of the goals must be reached. Deadlines are specified using the PDDL3 within constraint. The problems also have a number deadline constraints, specified with PDDL3 always-within constraint.

Complex Preferences This variant is similar to the previous, but has soft deadlines instead, encoded with preferences on the constraints. Each goal can have several (increasing) deadline, with different (increasing) penalties for missing them.” [?]

Problem Generator

Parameters

Generation

52 The FF Domain Collection (hoffmann)

WEBSITE

Description “This page was created by Joerg Hoffmann and Bernhard Nebel, to form a starting point for people who want to perform large-scale empirical studies in planning. The FF domain collection provides (randomized, where possible) generators for 20 STRIPS and ADL planning benchmark domains, including the examples used in both competitions. Below, we give, for each domain, information about the origin, made adaptations (if any), the parameters of the generator, and the randomization strategy. Downloads are available for the domain files and the C source code of the generators. Click here to download the whole package. The generators should be self-explanatory in terms of how to build them (makefiles are included), and how to run them. In the less obvious cases, we have also included a README file. Contact us with questions.

Note: We do not make any claims whatsoever about the validity of the problem ranges generated, nor about the adequacy of the kind of problems that are generated within a domain. We have made an effort to imitate closely the examples known from published problem suites, and we have generally chosen the most obvious first-guess randomization strategy. Some of the generators, like the one for Assembly, are quite a hack. Nevertheless, we believe that the generators form an invaluable tool for experimentation — they have definitely done so in our own experiments. Please contact us with any comments or suggestions.“ [HN06]

52.1 Assembly

Origin Drew McDermott. Used in the AIPS-1998 competition.

Adaptions None.

Description Typed ADL domain using complex pre- and effect-conditions, as well as conditional effects. Assemble one goal item by repeatedly putting together several items that are part of the same meta-item. The part-of relation defines a tree, where the root is the goal item and the leafs are the base items that do not need to be assembled. Any item can need a resource (like an oven) during the process of its being assembled. It may be that a “transient” part has to be assembled somewhere and removed afterwards, that parts of the same meta-item need to be assembled in a specified order (obeying assemble order constraints) and that some other part needs to be assembled before the transient part is removed again (obeying remove order constraints).

Problem Generator

Parameters

- d depth of part-of tree
- m maximal number of sons to any node in the tree
- h probability that a node has any sons
- n number of distinct resources

- r probability that a non-base node requires a resource
- t probability that an item is transient part for any higher-in-tree item
- a probability that a pair of items has an assemble order constraint
- o probability that an item has a remove ordering constraint with a transient part

Generation Create a tree of depth $-d$. Nodes have sons with probability $-h$. If they do have sons, then a random number between 1 and $-m$, biased to be lower the deeper the node is in the tree. With probability $-r$, the node needs a random resource. An item is assigned a transient part relation to any item in a higher tree level with probability $-t$, parts or transient parts of the same item have an assemble ordering constraint with probability $-a$, and for a transient part relation (A, B) the parts of B are given a remove ordering constraint to A with probability $-o$. Cycles in the assemble and remove order constraints are avoided by arbitrarily ordering the respective items, and allowing constraints only between pairs A and B with A \prec B.

52.2 Blocksworld-3ops

Origin Goes back to the annals of AI, seems to be first mentioned in Terry Winograd's PhD thesis 1972. In a 1974 planning paper by Gerald Sussman. Taken from the IPP domain collection.

Adaptions None.

Description Classical untyped STRIPS domain, where stackable blocks need to be re-assembled on a table with unlimited space. Representation uses 3 operators, moving a block from the table to another block, a block from another block to the table, or a block from a block to another block. Semantically, the representation does not use a robot arm, in difference to the 4 operator representation below. The initial state specifies a complete state, the goal state specifies only the on relations required between any two blocks.

Problem Generator

Parameters Number of blocks.

Generation Uses random blocksworld state generator provided by John Slaney and Sylvie Thiebaux. Simply translates two such states into PDDL, and prints them out as the initial state and the goal state, where the latter state has all facts removed except the on relations between blocks.

52.3 Blocksworld-4ops

Origin See above. Used in the AIPS-2000 competition. Taken from the IPP domain collection.

Adaptions None.

Description Like above, but uses a robot arm that can be used for stacking a block onto a block, unstacking a block from a block, putting down a block, or picking up a block.

Problem Generator

Parameters Number of blocks.

Generation Like above.

52.4 Briefcaseworld

Origin First mentioned by Edwin Pednault. Taken from the IPP domain collection.

Adaptions None.

Description Typed classical ADL domain, using conditional effects. Transport a number of objects from their start- to their goal-locations, using a briefcase. Each location is accessible from each other location, objects can be put into the briefcase or taken out of the briefcase. When a move is made between locations, then all objects inside the briefcase are also moved, which is encoded by a conditional effect.

Problem Generator

Parameters -o number of objects

Generation randomly distribute the start locations of all objects and the briefcase over -o + 1 locations. Do the same for the goal locations.

52.5 Ferry

Origin ?. Taken from the IPP domain collection.

Adaptions Sail operator modified so that moves can onENERATOR.tar.gz

52.6 Gripper

Origin Jana Koehler. Used in the AIPS-1998 competition.

Adaptions None.

Description Untyped STRIPS domain. Given a robot with two gripper hands, transport a number of balls from a room A to another room B.

Problem Generator

Parameters -n number of balls.

Generation No randomization. Place all balls in room A and require them to be in B instead.

52.7 Hanoi

Origin ?. Taken from the IPP domain collection.

Adaptions None.

Description Untyped STRIPS encoding of the well-known Towers of Hanoi problem.

Problem Generator

Parameters -n number of discs.

Generation No randomization.

52.8 Logistics

Origin First version by Manuela Veloso, AIPS-1998 version created by Bart Selman and Henry Kautz. Used in both the AIPS-1998 and AIPS-2000 competitions.

Adaptions None.

Description Classical untyped STRIPS domain. Transport packages within cities via trucks, and between cities via airplanes. Locations within a city are directly connected (trucks can move between any two such locations), and so are the cities. In each city there is exactly one truck, each city has one location that serves as an airport.

Problem Generator

Parameters

-c number of cities

-s size of each city, i.e. number of locations within cities

-p number of packages

-a number of airplanes

Generation Place trucks randomly within their cities, place airplanes randomly at airports. Distribute start and goal locations of packages randomly over all locations.

52.9 Miconic-ADL

Origin Jana Koehler. Used in the AIPS-2000 competition.

Adaptions None.

Description Typed ADL domain using complex preconditions and conditional effects. Transport a number of passengers with an elevator from their origin- to their destination floors. Obey several restrictions: some passengers must be transported directly, the vips shall be served first, some must be transported non-stop, some must be attended by others, some groups of people must not meet each other, some people do not have access to certain floors. When the lift stops at some floor, all passengers waiting there get in, and all passengers wanting to go there get out, by a conditional effect.

Problem Generator

Parameters

- f number of floors
- p number of passengers
- u percentage of passengers with direct transportation
- v percentage of vips
- g percentage of non-stop passengers
- n percentage of passengers that must be attended
- a percentage of passengers that can attend the above type
- A percentage of passengers in conflict group A
- B percentage of passengers in group B, which must not meet the above group
- N percentage of people with no access to some floors
- F percentage of of floors not to be accessed by those

Generation Original generator used in the AIPS-2000 competition. Make sure that the specified percentage values are all met by randomly assigning types to the passengers (a passenger can have several types). Randomly assign origin and destination floors to all passengers, considering several heuristics to help problem becoming solvable (like not placing conflicting people at the same origin floor etc.).

52.10 Miconic-SIMPLE

Origin Jana Koehler. Used in the AIPS-2000 competition.

Adaptions None.

Description Typed ADL domain using conditional effects. Like above, but without any additional constraints. The conditional effects make sure that all waiting passengers get in or out.

Problem Generator

Parameters

- f number of floors
- p number of passengers

Generation Original generator used in the AIPS-2000 competition. Simply distribute origin and destination floors at random.

52.11 Miconic-STRIPS

Origin Jana Koehler. Used in the AIPS-2000 competition.

Adaptions None.

Description Typed STRIPS domain. Like above, but with explicit control over the passengers that get in or out of the lift.

Problem Generator

Parameters

- f number of floors
- p number of passengers

Generation Original generator used in the AIPS-2000 competition. Simply distribute origin and destination floors at random.

52.12 Movie

Origin Corin Anderson. Used in the AIPS-1998 competition.

Adaptions None.

Description Untyped STRIPS domains. Buy one each out of five types of snacks, then rewind the movie and reset the counter.

Problem Generator

Parameters -n number of snacks of each type

Generation No randomization.

52.13 Mprime

Origin Drew McDermott. Used in the AIPS-1998 competition.

Adaptions Translated all predicate names to get a more intuitive notation. Operator for passing on fuel from one location to another could, in the original version, be instantiated with the same location as origin and destination city, which caused the amount of fuel in that city to increase one unit. Changed that such that origin and destination cities must be different.

Description Typed STRIPS domain; the name results from Mystery' (see below). Logistics variant where trucks move on a map of locations. Additionally, trucks have only limited transportation capacity, and there are constraints on the amount of fuel. Each location has initially a certain amount of fuel available. Moving a truck away from a location decreases the amount of fuel at that location by one. If a location has more than one fuel item, then it can pass a fuel item over to a different location.

Problem Generator

Parameters

- l number of locations
- f maximal amount of fuel at a location
- s maximal amount of transportation capacity (space)
- v number of trucks (vehicles)
- c number of cargos

Generation Create a simple map of -l locations such that location i is linked to location $i+1$, and location -l is linked to location 1. Randomly assign transportation capacity between 1 and -s to all vehicles, and fuel between 0 and -f to all locations. Distribute cargo origin and destination locations randomly over all locations, likewise for the starting locations of the vehicles.

52.14 Mystery

Origin Drew McDermott. Used in the AIPS-1998 competition.

Adaptions Translated all predicate names to get a more intuitive notation.

Description Typed STRIPS domain; the name is because the original specified the semantics in a disguised manner by using unintuitive names for the predicates and constants. The domain is the same like the Mprime domain above, except that there is no way of passing on fuel between locations.

Problem Generator

Parameters

- l number of locations
- f maximal amount of fuel at a location
- s maximal amount of transportation capacity (space)
- v number of trucks (vehicles)
- c number of cargos

Generation Exactly like in Mprime.

52.15 Schedule

Origin One variation appears in the Prodigy collection by Manuela Veloso. Prepared for the AIPS-2000 competition by Fahiem Bacchus.

Adaptions None.

Description Typed ADL domain using conditional effects. Encodes a simple Scheduling kind of problem where a number of objects need to be processed using a collection of machines. Possible actions are polishing, punching holes, painting etc. All actions need uniform time, which is modelled by a do-time-step operator. If that operator is applied, then all busy machines are no longer busy, and all scheduled objects are no longer scheduled—this is also an example of the kind of conditional effects that are used in the representation.

Problem Generator

Parameters (without significant changes to the domain, parameters -s to -o can not be arbitrarily increased)

- p number of objects (parts)
- s number of shapes (maximal 3: cylindrical, circular, oblong)
- c number of colors (maximal 4)
- w number of different widths of holes (maximal 3)
- o number of orientations of holes (maximal 2)
- Q probability that a part needs to be made cylindrical
- W probability that a part is initially coloured
- E probability that a part needs to be coloured
- R probability that a part has a hole initially
- T probability that a part needs to have a hole
- Y probability that a part needs to have a specific surface condition

Generation All parts are given a random initial shape and surface condition. With the respective probability, they are given random colours or holes. In the goal state, they need, with the respective probabilities, to be cylindrical (which is the only shape that can be produced by the machines), to be randomly coloured, to have a random hole, and to have a random goal surface condition.

52.16 Tsp

Origin Obtained from Maria Fox and Derek Long.

Adaptions None.

Description Untyped STRIPS domain. Extremely simple version of TSP. The locations are connected by a complete graph, i.e. each location is accessible from each other location. The edges all have equal cost—one moving operation—and the goal is simply to have all locations visited. An optimal solution simply visits all locations once in an arbitrary ordering.

Problem Generator

Parameters -n number of locations

Generation No randomization.

52.17 Tyreworld

Origin Stuart Russel. Adaption for multiple tyres by Jana Koehler. Taken from the IPP domain collection.

Adaptions None.

Description Typed STRIPS domain. Replace a flat tyre with a spare one. This involves fetching the tools (wrench, jack, pump) from the boot, undoing the nuts on the flat tyre, jacking up the (appropriate) hub(s), removing the tyre, doing up the spare one, etc. Adapted for several tyres by simply increasing the number of flat tyres to be replaced.

Problem Generator

Parameters -n number of tyres.

Generation No randomization.

53 UCPOP Benchmarks (strict)

53.1 Art

Origin TODO

Adaptions None.

Description Multiple variants art, art-1d, art-1d-rd, art-md, art-md-ns, art-md-ns-rd, art-md-rd [Citation Unknown]

Problem Generator

Parameters

Generation

53.2 blocks-world

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.3 briefcase

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.4 d1s1

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.5 ds

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.6 eight

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.7 ferry

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.8 fridge

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.9 homeowner

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.10 ipp-domains

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.11 logistics

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.12 meet-pass

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.13 molgen

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.14 monkey

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.15 montlake

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.16 morris

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.17 mystery

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.18 **occam**

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.19 **safety-domain**

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.20 **tire-world**

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.21 **trains**

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.22 transplan

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.23 travel

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.24 truckworld

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

53.25 woodshop

Origin TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

54 Other Domains

54.1 Link Chain

Origin Manuala Veloso?? TODO

Adaptions None.

Description [Citation Unknown]

Problem Generator

Parameters

Generation

54.2 Sodor

Origin Maria Fox?? TODO

Adaptions None.

Description [HDH⁺99]

Problem Generator

Parameters

Generation

54.3 Storage Tek (STEK)

54.4 TEMPLATE

Origin TODO

Adaptions None.

Description [HDH⁺99]

Problem Generator

Parameters

Generation

54.5 Memon

WEBSITE

Origin TODO

Adaptions None.

Description [MemXX]

Problem Generator

Parameters

Generation

54.6 PSPLib

Patrick Haslum has written a conversion program called chopshop that converts MRCPS (multi-mode resource Constrained Project Scheduling) problems from the PSPLIB into PDDL. The Chopshop program is shipped with the HSP* code.

References

- [AIP98] AIPS-98 International Planning Competition Committee. PDDL : the planning domain definition language. Technical report, Yale Center for Computational Vision and Control, October 1998. The AIPS-98 committee consisted of: M. Ghallab and A. Howe and C. Knoblock and D. McDermott and A. Ram and M. Veloso and D. Weld and D. Wilkins.
- [Ger06] Alfonso Gerevini. Ipc-5 home page. Web Page, June 12 2006. last accessed January 5, 2007.
- [HDH⁺99] Adele Howe, Eric Dahlman, C. Hansen, A. vonMayrhauser, and M. Scheetz. Exploiting competitive planner performance. In *Proc. of ECP-99*, Durham, England, September 1999.
- [HE03] Joerg Hoffmann and Stefan Edelkamp. IPC-4 home page. Web Page, July 17 2003. last accessed January 5, 2007.
- [HE04] Jorg Hoffmann and Stefan Edelkamp. Towards realistic benchmarks for planning: the domains used in the classical part of IPC-4 (extended abstract)., 2004.
- [HN06] Joerg Hoffmann and Bernhard Nebel. FF domain collection. Web Page, September 20 2006. Last accessed January 5, 2007.
- [LF03] Derek Long and Maria Fox. Ipc home: The 2002 competition. Web Page, November 25 2003. last accessed January 5, 2007.

- [McD98] Drew McDermott. Planning competition results. FTP Website, 1998. Last accessed January 3, 2007.
- [MemXX] Memon. Memon problems. Incomplete Reference, XX XX. XXXX.

Part V

DOMAIN ANALYSIS TOOLS

55 TIM (STAN's Type Interface Module)

[FL98]

56 RIFO (Removing irrelevant facts and operators from planning problems)

RIFO WEBSITE

Description “To limit search to those facts and operators only, which are indeed necessary to generate a solution plan, most common planning systems start searching from the goals. Graphplan and IPP, however, build a “planning graph” starting from the initial facts. This has been a very successful approach, but can lead to ineffective graph expansion and search processes if the planning task specification contains irrelevant information.

RIFO tries to determine such irrelevant information (ground operators and initial facts) using a “backchaining” process and removes them from the planning task. While trying to reach the goals recursively, sets of possibly relevant objects and initial facts are built. In order to prevent this procedure from becoming a complete planning process a simple heuristics is used: Try to achieve ALL goals (AND nodes) by backchaining and branching on different ground operators (OR nodes), which achieve one of the goals. The process is repeated recursively using operator preconditions and effect conditions as new goals, which generates a so called “fact generation tree”, in which potential conflicts between goals and operators are ignored.

Depending on the heuristic and union method chosen, different kinds of “possibility sets” of relevant objects and facts are created. These sets can be used in different ways to decide over relevance or irrelevance of ground operators and initial facts.” [Koe00b]

57 GAM (IPP's Goal Agenda Manager)

IPP-GAM WEBSITE

Description “The GAM option inside IPP implements a new approach to order conjunctive goals. It was developed by Jana Koehler and Joerg Hoffmann and is available in IPP 3.3. Today, GAM is further developed inside the FF planner developed by Joerg Hoffmann, which is based on heuristic forward search.” [Koe00a]

58 XGV 1.1

XGV WEBSITE

Description “The graphical interface for the inspection of planning graphs on SUN workstations. Requires certain Motif libraries, which are almost nowhere available. Here is XGV for PC under Linux (statically linked and compiled). XGV documentation.” [Koe]

59 PDB X.X

Quoted from the document doc/ucpop/pdb.html included with the repop distributions. **DEBUG: Locate code and further information for PDB.**

Description “The primary function of PDB is to allow the user to schedule planning problems to be run by the planner, record the actions taken by the planner, and then reconstruct that information in a meaningful way. Since UCPOP does extensive search in plan-space, PDB focuses on recording and presenting plan-space search trees generated by the planner.

PDB is an interactive application which makes the UCPOP planning process more accessible to the user by the use of graphical interfaces to the planning process. It is intended to be used in several ways: Domain Construction, Search Control Development, Planner Modification.” [Unkbd]

60 DISCOPLAN

[GS03]

A. Gerevini, L. K. Schubert, Discovering State Constraints in DISCOPLAN: Some New Results, Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000)

A. Gerevini, L. K. Schubert, Extending the Types of State Constraints Discovered by DISCOPLAN , Workshop on Analyzing and Exploiting Domain Knowledge for Efficient Planning, AIPS-2000

A. Gerevini, Inferring State Constraints as Control Knowledge for Domain-Independent Planning (extended abstract) Proceedings of the Workshop on Planning as Combinatorial Search, AIPS-1998

A. Gerevini, L. K. Schubert, Inferring State Constraints for Domain-Independent Planning, Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI-1998)

A. Gerevini, L. K. Schubert, Computing Parameter Domains as an Aid to Planning, Proceedings of the Third International Conference on Artificial Intelligence Planning Systems (AIPS-1996)

A. Gerevini, L. K. Schubert, Accelerating Partial Orders Planners: Some Techniques for Effective Search Control and Pruning, Journal of Artificial Intelligence Research - JAIR, 5, 1996

61 VAL - Automatic Validation Tool for PDDL

VAL WEBSITE [Str]

62 RedOp - Reduced Operator Sets (now called pddlcat)

REDOP WEBSITE

“RedOp is (sort of) a domain analysis tool for planning. It finds and removes redundant operator instances, as described in the [HJ00]. In the process, it also derives some static facts and uses these to simplify the domain and operators.

Note: RedOp is now (re-)implemented in the pddlcat tool, included in the HSP* planner package. The information below refers to the old implementation. “ [Has02]

References

- [FL98] M. Fox and D. Long. The automatic inference of state invariants in TIM. *JAIR*, Volume 9:367–421, 1998.
- [GS03] Alfonso Gerevini and Len Schubert. Discovering state constraints for planning: Dis-coplan. Technical Report 811, Dept. of Computer Science, University of Rochester, Rochester, USA, September 2003.
- [Has02] Patrik Haslum. Redop homepage. Web Page, July 2002. Last accessed January 10, 2007.
- [HJ00] P. Haslum and P. Jonsson. Planning with reduced operator sets. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling*. AAAI Press, 2000. See also [Has02].
- [Koe] Jana Koehler. IPP publications. Web Page. Last Access January 3, 2007.
- [Koe00a] Jana Koehler. IPP-GAM: Solving complex planning tasks through extraction of subproblems. Web Page, May 23 2000. Last accessed January 5, 2007.
- [Koe00b] Jana Koehler. IPP-RIFO. Web Page, May 23 2000. Last accessed January 5, 2007.
- [Str] Strathclyde Planning Group. Val, the automatic validation tool for PDDL, including PDDL3 and PDDL+. Web Page.
- [Unkbd] Unknown. *PDB Reference Manual*. UCPOP Development Team, tbd. Found document while looking into RePOP - its an old tool for UCPOP. Its current location is component-planners/repop/doc/ucpop-doc/pdb.html.

References

- [AIP98] AIPS-98 International Planning Competition Committee. PDDL : the planning domain definition language. Technical report, Yale Center for Computational Vision and Control, October 1998. The AIPS-98 committee consisted of: M. Ghallab and A. Howe and C. Knoblock and D. McDermott and A. Ram and M. Veloso and D. Weld and D. Wilkins.
- [AK01] J.L. Ambite and C.A. Knoblock. Planning by rewriting. *JAIR*, 15:207–261, 2001.
- [Ama03] Rob St. Amant. Planning resources. Web Page, July 9 2003. Last accessed January 3, 2006.
- [Bac00] Fahiem Bacchus. Aips-00 planning competition: The fifth international conference on artificial intelligence planning and scheduling systems. Web Page, 2000. Last Accessed January 3, 2007.
- [BG00] Blai Bonet and Hector Geffner. *Heuristic Search Planner 1.1 README*, June 21 2000. Included with the distribution for HSP 1.1.
- [CS07] A. I. Coles and A. J. Smith. Marvin: A heuristic search planner with online macro-action learning. *JAIR*, 28:119–156, 2007.
- [DK03] Minh Binh Do and Subbarao Kambhampati. Sapa: A multi-objective metric temporal planner. *JAIR*, 20:155–194, 2003.
- This planner is described in Section 33. This paper runs Sapa on problems 47.4.
- [FB05] Eugene Fink and Jim Blythe. Prodigy bidirectional planning. *Journal of Experimental and Theoretical Artificial Intelligence*, 17(3):161–200, 2005.
- [FL98] M. Fox and D. Long. The automatic inference of state invariants in TIM. *JAIR*, Volume 9:367–421, 1998.
- [Ger06] Alfonso Gerevini. Ipc-5 home page. Web Page, June 12 2006. last accessed January 5, 2007.
- [GNT04] Malik Ghallab, Dana Nau, and Paolo Traverso. *Automated Planning*. Morgan Kaufmann Publishers, May 2004.
- [GS99] Alfonso Gerevini and Ivan Serina. Fast planning through greedy action graphs. In *Proceedings of Sixteenth National Conference of Artificial Intelligence (AAAI-99)*, Orlando, Florida, USA, 1999. AAAI-MIT Press.
- [GS00] Alfonso Gerevini and Ivan Serina. Fast plan adaptation through planning graphs: Local and systematic search techniques. In *Proceedings of the 5th International Conference on Artificial Intelligence Planning Systems (AIPS-00)*, Breckenridge, Colorado, USA, 2000. AAAI Press.
- [GS01] Alfonso Gerevini and Ivan Serina. Lagrange multipliers for local search on planning graphs. In A. Nareyek, editor, *Proceedings of the ECAI workshop on Local Search for Planning and Scheduling*, number Lecture Notes in Artificial Intelligence 2148. Springer-Verlag, 2001.

- [GS03a] Alfonso Gerevini and Len Schubert. Discovering state constraints for planning: Discoplan. Technical Report 811, Dept. of Computer Science, University of Rochester, Rochester, USA, September 2003.
- [GS03b] Alfonso Gerevini and Ivan Serina. Planning as propositional CSP: from Walksat to local search for action graphs. *CONSTRAINTS*, 8:389–413, 2003.
- [GSS03] A. Gerevini, A. Saetti, and I. Serina. Planning through stochastic local search and temporal action graphs in LPG. *JAIR*, 20:239–290, 2003.
- [GSS04] Alfonso Gerevini, Alessandro Saetti, and Ivan Serina. Planning in PDDL2.2 domains with LPG-TD. In *Proc. of the 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04), International Planning Competition abstract*, Whistler, Canada, 2004.
- [GSS06] Alfonso Gerevini, Alessandro Saetti, and Ivan Serina. An approach to temporal planning and scheduling in domains with predicatable exogenous events. *JAIR*, 25:187–231, 2006.
- [GSSP04] Alfonso Gerevini, Alessandro Saetti, Ivan Serina, and Paolo Toninelli. LPG-TD: a fully automated planner for PDDL2.2 domains. In *Proc. of the 14th Int. Conference on Automated Planning and Scheduling (ICAPS-04) International Planning Competition abstracts*, Whistler, Canada, 2004.
- [GSSS03] Alfonso Gerevini, Ivan Serina, Alessandro Saetti, and Sergio Spinoni. Local search techniques for temporal planning in LPG. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS-03)*, Trento, Italy, 2003. AAAI Press.
- [Has02] Patrik Haslum. Redop homepage. Web Page, July 2002. Last accessed January 10, 2007.
- [Has04] Patrik Haslum. Hsp* homepage. Web Page, July 2004. Last accessed January 9, 2007.
- [HD02] Adele Howe and Eric Dahlman. A critical assessment of benchmark comparison in planning. *JAIR*, 17:1–33, July 2002.
- [HDH⁺99] Adele Howe, Eric Dahlman, C. Hansen, A. vonMayrhauser, and M. Scheetz. Exploiting competitive planner performance. In *Proc. of ECP-99*, Durham, England, September 1999.
- [HE03] Joerg Hoffmann and Stefan Edelkamp. IPC-4 home page. Web Page, July 17 2003. last accessed January 5, 2007.
- [HE04] Jorg Hoffmann and Stefan Edelkamp. Towards realistic benchmarks for planning: the domains used in the classical part of IPC-4 (extended abstract)., 2004.
- [Hel06] Malte Helmert. The fast downward planning system. *Journal of Artificial Intelligence Research*, 26:191–246, 2006.
- [HJ00] P. Haslum and P. Jonsson. Planning with reduced operator sets. In *Proc. 5th International Conference on Artificial Intelligence Planning and Scheduling*. AAAI Press, 2000. See also [Has02].

- [HK99] J. Hoffmann and J. Koehler. A new method to query and index sets. In *Proc. of IJCAI-99*, 1999.
- [HN06] Joerg Hoffmann and Bernhard Nebel. FF domain collection. Web Page, September 20 2006. Last accessed January 5, 2007.
- [Hof03] J. Hoffmann. The Metric-FF planning system: Translating “ignoring delete lists” to numeric state variables. *JAIR*, 20:291–341, 2003.
- [KH98] Jana Koehler and Joerg Hoffmann. Planning with goal agendas. Technical report, Institute for Computer Science, Albert Ludwigs University, 1998.
- [KNHD97a] Jana Koehler, Bernhard Nebel, Joerg Hoffmann, and Yannis Dimopoulos. Extending planning graphs to an ADL subset. In *Proc. ECP-97*, pages 273–285, London, UK, 1997. Springer-Verlag.
- [KNHD97b] Jana Koehler, Bernhard Nebel, Joerg Hoffmann, and Yannis Dimopoulos. Extending planning graphs to an ADL subset. Technical Report 88/97, Institute for Computer Science, Albert Ludwigs University, 1997.
- [Koe] Jana Koehler. IPP publications. Web Page. Last Access January 3, 2007.
- [Koe98] J. Koehler. Solving complex planning tasks through extraction of subproblems. In *Proc. of AIPS-98*, pages 62–69. AAAI Press, 1998.
- [Koe99] J. Koehler. RIFO within IPP. Technical Report 126/99, Institute for Computer Science, Albert Ludwigs University, Am Flughafen 17, 79110 Freiburg, Germany, 1999.
- [Koe00a] Jana Koehler. IPP-GAM: Solving complex planning tasks through extraction of subproblems. Web Page, May 23 2000. Last accessed January 5, 2007.
- [Koe00b] Jana Koehler. IPP-RIFO. Web Page, May 23 2000. Last accessed January 5, 2007.
- [Koe06] Jana Koehler. IPP homepage. Web Page, October 15 2006. Last accessed September 11, 2007.
- [KS92] Henry Kautz and Bart Selman. Planning as satisfiability. In *Proceedings ECAI-92*, 1992.
- [KSH06] Henry Kautz, Bart Selman, and Joerg Hoffmann. Satplan: Planning as satisfiability. In *Abstracts of the 5th International Planning Competition, ICAPS-06.*, 2006.
- [LF] Derek Long and Maria Fox. Stan: An spg planning system. Web Page. Last accessed January 8, 2007.
- [LF99] D. Long and M. Fox. Efficient implementation of the plan graph in STAN. *JAIR*, 10:87–115, 1999.
- [LF03] Derek Long and Maria Fox. Ipc home: The 2002 competition. Web Page, November 25 2003. last accessed January 5, 2007.

- [Lin01] Fangzhen Lin. A planner called R. *Artificial Intelligence*, 22(3):73–76, Fall 2001.
- [Liu04] Donghong Liu. PLANET planning database: Online research database. Web Page, March 9 2004. Last accessed January 9, 2004.
- [McD98] Drew McDermott. Planning competition results. FTP Website, 1998. Last accessed January 3, 2007.
- [McD05] Drew McDermott. *The Opt and Optop API*. Yale Computer Science Department, 0.9 edition, November 2005.
- [MemXX] Memon. Memon problems. Incomplete Reference, XX XX. XXXX.
- [NDK97] Bernhard Nebel, Yannis Dimopoulos, and Jana Koehler. Ignoring irrelevant facts and operators in plan generation. In *Proceedings of the 4th European Conference on Planning*, volume 1348 of LNAI, pages 338–350. Springer, 1997.
- [NK64] X. Nguyen and S Kambhampati. Reviving partial order planning. In B. Nebel, editor, *IJCAI-01*, Seattle, WA., 459-464. Morgan Kaufmann Publishers. Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.
- [Pla] PlanSIG. Plansig: Planners and schedulers. Web Page.
- [PW92] J. S. Penberthy and D. Weld. UCPOP: A sound, complete, partial-order planner for ADL. In *Proceedings 3rd International Conference on Principles of Knowledge Representation and Reasoning (KR-92)*, Cambridge, MA, October 1992.
- [PW94] J.S. Penberthy and D. Weld. Temporal planning with continuous change. In *AAAI-94*, July 1994.
- [Ref03] Ioannis Refanidis. Grt planner - home page. Web Page, June 2003. Last accessed January 3, 2007.
- [RN02] Stuart Russell and Peter Norvig. *Introduction to Artificial Intelligence*. Prentice Hall, second edition, 2002.
- [RV00] I. Refanidis and I. Vlahavas. Exploiting state constraints in heuristic state-space planning. In *5th International Conference on Artificial Intelligence Planning and Scheduling Systems (AIPS-2000)*, Breckenridge, Colorado, April 2000.
- [RV01] I. Refanidis and I. Vlahavas. The grt planner: Backward heuristic construction in forward state-space planning. *Journal of Artificial Intelligence Research*, 15:115–161, 2001. The companion source code appendix includes the GRT Planner, AltAlt, FF, HSP, and STAN as well as a number of problems and data used to complete the paper.
- [RVT99] I. Refanidis, I. Vlahavas, and L. Tsoukalas. On determining and completing incomplete states in strips domains. In *IEEE International Conference on Information, Intelligence and Systems*, pages 289–296, Washington D.C., November 1-3 1999.
- [Sap06] Oscar Sapena. Simplanner publications. Web Page, July 3 2006. Last Accessed January 3, 2007.

- [SP96] David E. Smith and Mark A. Peot. Suspending recursion in causal link planning. In B. Drabble, editor, *Proceedings of the 3rd International Conference on Artificial Intelligence Planning Systems (AIPS-96)*, pages 182–190. AAAI Press, 1996.
- [Str] Strathclyde Planning Group. Val, the automatic validation tool for PDDL, including PDDL3 and PDDL+. Web Page.
- [unk] The ucpop planner. Web Page.
- [Unk06] Unknown. Daml.org. Web Page, January 29 2006. Last Accessed January 9, 2007.
- [Unkbd] Unknown. *PDB Reference Manual*. UCPOP Development Team, tbd. Found document while looking into RePOP - its an old tool for UCPOP. Its current location is component-planners/repop/doc/ucpop-doc/pdb.html.
- [VBC99] P. Van Beek and X. Chen. Cplan: a constraint programming approach to planning. In *Proc. National Conference on Artificial Intelligence*, pages 585–590. AAAI Press/MIT Press, 1999.
- [VCP⁺95] Manuela Veloso, Jaime Carbonell, Alicia Perez, Daniel Borrajo, Eugene Fink, and Jim Blythe. Integrating planning and learning: The PRODIGY architecture. *Journal of Experimental and Theoretical Artificial Intelligence*, 7(1):81–120, 1995.
- [Vid04] V. Vidal. A lookahead strategy for heuristic search planning. In *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS-04)*, pages 150–159, Whistler, CA, June 2004.
- [VTBV03] D. Vrakas, G. Tsoumakas, N. Bassiliades, and I. Vlahavas. Learning rules for adaptive planning. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS03)*, pages 82–91, Trento, Italy, June 9-13 2003.
- [VTBV05] D. Vrakas, G. Tsoumakas, N. Bassiliades, and I. Vlahavas. *Intelligent Techniques for Planning*, chapter Machine Learning for Adaptive Planning, pages 90–120. Idea Group, 2005.
- [WAS98] Daniel S. Weld, Corin R. Anderson, and David E. Smith. Extending Graphplan to handle uncertainty and sensing actions. In *Proc. AAAI-98*, pages 897–904, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.
- [Wel99] Daniel S. Weld. Recent advances in AI planning. *AI Magazine*, 20(2):93–123, 1999.
- [YS03] H.L.S. Younes and R.G. Simmons. VHPOP: Versatile heuristic partial order planner. *JAIR*, 20:405–430, 2003.