Proceedings of the Thirty-Sixth International Conference on Automated Planning and Scheduling (ICAPS 2026)

Planner Museum: Evaluating Classical Planners Over Time

Arnaud Lequen, Oliver Joergensen, Windy Phung, Elliot Gestrin, Damien Van Meerbeeck, Markus Fritzsche, Dominik Drexler, Jendrik Seipp

Linköping University, Sweden
firstname.lastname@liu.se

Abstract

The International Planning Competition (IPC) has been running since 1998, providing a standardized platform on which different planners can compete on equal footing. Over the years, the competition has showcased diverse approaches to planning and has thus become a record of the paradigm shifts the field has undergone. In this paper, we introduce the Planner Museum, a project that collects distinguished planning systems from past competitions and enables them to run on modern computers. By evaluating these historical planners alongside modern ones on challenging benchmark problems from multiple IPCs, we quantify the rate of improvement of state-of-the-art planners over the past 28 years and show that progress has been steady.

Introduction

Ever since 1998, the International Planning Competition (IPC) has served as a showcase for progress in automated planning. By allowing planners to compete on a standardized set of benchmarks, it has documented the field's progress over the past 28 years.

In the 2023 edition of the IPC, LAMA (Richter and Westphal 2008), a planner whose first version dates back to 2008, ranked fourth out of 22 in the Satisficing Track. More surprisingly, it outperformed all planners competing in the Agile Track (Taitler et al. 2024). This raises a natural question: have we made conceptual progress in suboptimal classical planning over the past few decades?

To investigate this question, we collect planners from across the IPC's history and assess their performance. Our work is inspired by the SAT Museum (Biere et al. 2023), a similar effort for SAT solvers, as well as by related analyses from the SAT community (Dutertre 2020; Kochemazov, Ignatiev, and Marques-Silva 2021; Audemard, Paulevé, and Simon 2020; Fichte, Hecher, and Szeider 2020). Like these efforts, our work serves as a sanity check for the field of automated planning. Our contribution is twofold.

The first contribution is a digital museum of distinguished IPC planners. It includes source code for 29 planners, patched when necessary, together with an Apptainer container for each planner to support easy use on modern hardware (Singularity Developers 2021). The second contribution is an evaluation of their performance on a benchmark set spanning the entire history of the IPC. In this comparison, instance difficulty is scaled so as not to favor planners that perform well only on particular problem types. Using modern hardware and coverage as our primary metric, we show that, despite modest beginnings in which Fast Forward (Hoffmann 2000) dominated, different planners won successive competitions, with performance increasing steadily since 2008.

Background

Automated planning is a model-based approach to sequential decision making. Many systems have been developed to address this problem, such as the Stanford Research Institute Problem Solver, which dates to 1971 (Fikes and Nilsson). Since then, the field has produced a large body of research and numerous textbooks have surveyed its major approaches (Ghallab, Nau, and Traverso 2004; Geffner and Bonet 2013).

The first IPC took place in 1998. It was enabled by the introduction of the Planning Domain Definition Language (PDDL) (McDermott et al. 1998), which provided a standard language for expressing planning problems. From 1998 through 2023, the IPC was held every two to three years, with longer gaps between more recent editions. Its scope has broadened since 1998, allowing not only classical planners to compete, but also probabilistic, temporal, and learning planners. In the classical setting, multiple tracks have emerged, such as the Satisficing, Agile, and Optimal tracks. In this work, we focus on domain-independent, non-optimal classical planning, which uses the simplest planning formalism. This choice allows planners developed before the introduction of these tracks to be compared fairly. We therefore consider only the number of instances solved within time and memory constraints, i.e., the *coverage*.

Methodology

Planner Selection. While early IPCs had only a few entrants, recent competitions feature more than 20 planners in each non-optimal classical track. We focus on distinguished planners according to the following criteria.

When a competition has a single track, which is the case through 2008, we pick the top three planners according to the ranking provided by the organizers. When a competition

Year	1998			2000				2002		2004		2006	2008			2011		2014				2018			2023				
	BB2	HSP	IPP	FF	HSP2	MIPS	SysR	LPG	SimP	FD	FDD	MIPSXXL	C3	FFSA	LAMA08	FDSS11	LAMA11	Probe	Merc	MpC	YAHSP	FDRemix	LAPKT	Saar	ANS	DecS	FDSS23	Levitron	Maidu
1998	3	5	2	68	41	14	30	38	5	57	58	0	48	61	61	73	63	53	66	48	55	74	52	66	57	84	**89**	82	84
2000	4	2	4	62	57	2	61	40	21	49	62	7	63	50	70	76	79	75	79	45	79	80	64	78	69	72	84	**85**	**85**
2002	2	9	4	95	33	13	62	47	100	79	83	1	61	71	105	89	102	80	109	64	79	101	89	115	87	103	101	**116**	102
2004	1	0	1	37	37	0	0	7	46	61	77	1	46	43	26	77	66	71	65	36	76	75	84	82	83	67	77	**88**	83
2006	0	0	0	40	10	2	60	2	4	16	27	4	29	23	52	39	62	35	64	23	33	59	51	**78**	41	57	67	69	65
2008	23	21	26	119	90	33	42	47	38	90	100	46	121	99	118	125	130	110	130	91	105	124	93	137	113	116	133	**145**	134
2011	3	0	2	48	19	2	11	6	13	23	28	1	41	57	101	89	122	107	143	50	83	140	94	**149**	93	103	141	144	134
2014	6	0	0	15	12	1	3	8	13	4	23	2	19	17	32	86	67	48	64	5	38	99	74	113	69	107	**114**	108	107
2018	0	0	1	53	0	0	0	0	1	0	0	1	23	28	79	83	90	57	99	18	9	96	60	108	97	92	104	**116**	101
Total	**42**	**37**	**40**	**537**	**299**	**67**	**269**	**195**	**241**	**379**	**458**	**63**	**451**	**449**	**644**	**737**	**781**	**636**	**819**	**380**	**557**	**848**	**661**	**926**	**709**	**801**	**910**	**953**	**895**

Table 1: Coverage obtained by each planner, summed over the domains introduced in each IPC year. Bold numbers indicate the highest coverage for the corresponding year. For space reasons, coverage is aggregated by IPC year; per-domain results are available in the online repository (Lequen et al. 2026a).

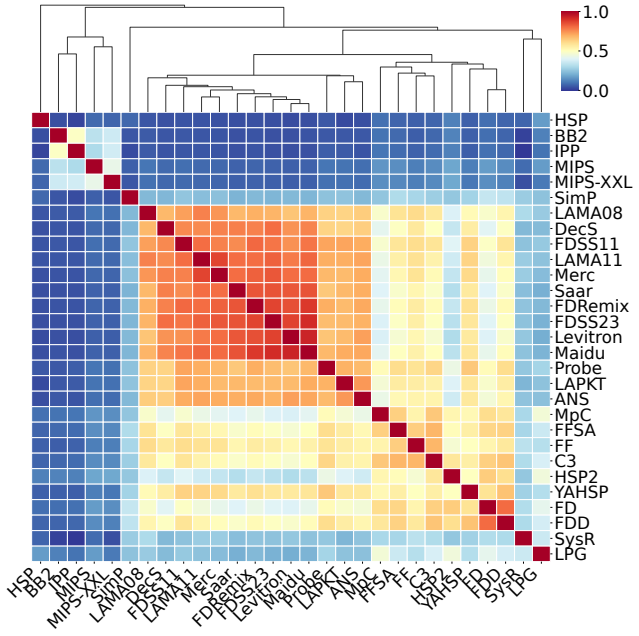

Figure 1: Similarity between planners, computed using the Jaccard index. Higher values indicate greater similarity between planners. The dendrogram groups planners with the highest similarity scores: the lower a join appears in the dendrogram, the more similar the planners involved are.

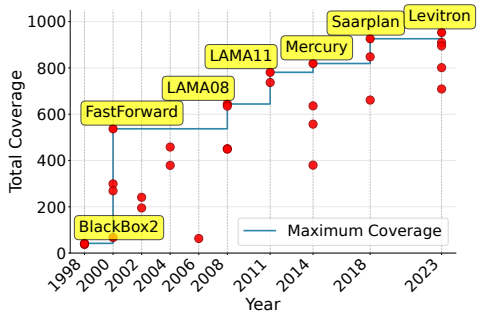

Figure 2: Maximum coverage attained by a single planner released at or before a given IPC. Each red dot represents a planner. Larger red dots indicate planners whose coverage exceeds that of all previously released planners. The blue line traces the frontier. Each vertical line marks a year in which an IPC was held.

has both an Agile Track and a Satisficing Track, we apply the same criterion to both tracks separately. If a planner is distinguished in both tracks, we keep the Agile version. For the first IPC in 1998, we include all planners whose code we could obtain. We exclude planners that already distinguished themselves in an earlier IPC or in a different track, in order to highlight algorithmic progress rather than code optimization alone. The exceptions are LAMA, whose 2011 version features algorithmic improvements over the 2008 version, including a different search sequence (Richter, Westphal, and Helmert 2011), and Fast Downward Stone Soup (FDSS), a portfolio planner. We include both FDSS versions, from 2011 and 2023, because they rely on different underlying planners.

When a planner's source code was unavailable online, we contacted the original authors. Since pre-2008 IPCs did not require code disclosure, some planners were excluded because their source code remained inaccessible. Planners in the SGPlan family were excluded because the IPC 2008 entry competed unfairly by relying on domain-specific knowledge (as confirmed by our inspection of the source code).

Planner Adaptation. To ensure consistent evaluation, we used the same operating system (Ubuntu 24.04 LTS) and compilers across all planners whenever possible. This required us to adapt older planners that had been developed for earlier software stacks. For instance, newer GCC versions reject code that older versions merely warned about, so we had to adjust the compilation flags for many historical planners.

As PDDL has undergone multiple revisions over the years, some domains in our benchmark set use constructs that are not supported by earlier planners, such as `:equality`. Although we tried to ensure that our benchmark set uses the simplest possible constructs, a few more elaborate ones re-

mained. For planners that do not support them, we added a preprocessor based on Loki (Drexler 2024) that compiles these features away.

Reproducibility. For each planner in our collection, we wrote an Apptainer recipe that generates standardized images. These images provide a uniform interface for all planners. The experimental artifacts are available online (Lequen et al. 2026b,a), including source code for all planners, Apptainer files and benchmarks. To conduct the experiments, we used Downward Lab (Seipp et al. 2017).

Benchmark Set. Previous IPCs featured interesting domains, but the original instances are often too simple to challenge modern planners. Fortunately, many domains come with generators that can create larger instances (Seipp, Torralba, and Hoffmann 2022). To preserve the discriminative power of older domains, we use the Autoscale benchmarks (Torralba, Seipp, and Sievers 2021), which suitably adjust instance sizes. We also remove operator cost information, when applicable, to ensure compatibility with older planners. Our benchmark set contains 42 domains spanning the entire history of the IPC, except for those introduced in 2023, which were not part of the Autoscale benchmarks.

Evaluation Metrics. Choosing evaluation metrics for cross-era comparison requires careful consideration, as IPC metrics have evolved over time. The first competitions measured planner coverage, sometimes across multiple PDDL formalisms, which favored planners that supported a broader range of language constructs. In such cases, as with IPC 2002, we recomputed the ranking using the original results, restricted to STRIPS domains. Other early competitions, such as the 2004 and 2006 editions, counted the number of domains where a planner outperformed all competitors.

Since 2011, the evaluation approach has been more standardized, with planners typically optimizing one of two metrics: the agile score, which quantifies how quickly a planner can find any solution, and the satisficing score, introduced in 2008, which measures plan quality. As a consequence, many early planners were not specifically optimized for runtime efficiency or plan quality. For this reason, we use coverage as our primary evaluation metric, as it is the only metric that all planners aimed to optimize, regardless of the competition in which they participated.

This metric is not perfect, however. Some planners trade coverage for better agile or satisficing scores. For instance, DecStar, the winner of the 2023 IPC Agile Track, sacrifices landmark support for faster computation, reducing coverage in some domains.

Selected Planners

We now present the planners selected for our experimental evaluation, organized chronologically by the International Planning Competition in which they were distinguished.

From the 1998 IPC, we select *BlackBox2* (BB2) (Kautz and Selman 1999), a Graphplan-based planner that compiles planning problems into SAT; *HSP* (Bonet and Geffner 2001b), a heuristic search planner using additive heuristics, and *IPP* (Koehler et al. 1997), a Graphplan-based planner.

From the 2000 IPC, we select *Fast Forward* (FF) (Hoffmann and Nebel 2001), a heuristic search planner that introduces the eponymous heuristic; *HSP2* (Bonet and Geffner 2001a), which builds upon HSP and uses more informative heuristic functions; *MIPS* (Edelkamp 2001), a planner based on model-checking techniques; and *System R* (SysR) (Lin 2001), a regression-based planner written in Prolog that tries to achieve one goal at a time.

From the 2002 IPC, we select *LPG* (Gerevini, Saetti, and Serina 2003), a local search planner based on planning graphs; *SimPlanner* (SimP) (Onaindia et al. 2001), a hill-climbing-based planner that combines relaxation-based heuristics to achieve goals separately.

From the 2004 IPC, we select *Fast Downward* (FD) (Helmert 2006b), a heuristic search planner with causal graph analysis, which works on a finite-domain representation of the planning task; and its variant *Fast Diagonally Downward* (FDD) (Helmert 2006a), which uses multi-heuristic best-first search.

From the 2006 IPC, we select *MIPS-XXL* (Edelkamp, Jabbar, and Nazih 2006), a version of MIPS that handles larger instances using external memory.

From the 2008 IPC, we select *C3* (Lipovetzky, Ramirez, and Geffner 2008), a two-tiered forward search planner that relies on incomplete backtracking in a pruned state space, and on search with the additive heuristic; *FFSA* (Keyder and Geffner 2008), a Fast Forward variant with the additive heuristic modified to propagate sets of actions representing relaxed plans; and *LAMA* (LAMA08) (Richter and Westphal 2008), which uses landmark heuristics and FF.

From the 2011 IPC, we select *Fast Downward Stone Soup 2011* (FDSS11) (Helmert et al. 2011), a portfolio planner; and *LAMA* (LAMA11) (Richter, Westphal, and Helmert 2011), an updated version of LAMA.

From the 2014 IPC, we select *Probe* (Lipovetzky et al. 2014), a planner that launches probes: depth-first searches from a given state that either find a goal or fail; *Mercury* (Merc) (Katz and Hoffmann 2014), which is based on red-black planning; *Madagascar-pC* (MpC) (Rintanen 2014), a SAT-based planner adapted to larger instances, with a SAT-solving heuristic tailored to planning; and *YAHSP3* (Vidal 2014), a heuristic search planner that embeds a lookahead policy based on an analysis of relaxed plans.

From the 2018 IPC, we select *Fast Downward Remix* (FDRemix) (Seipp 2018), a precomputed sequential static portfolio of Fast Downward configurations; *LAPKT-BFWS* (LAPKT) (Ramirez, Lipovetzky, and Muise 2015), a best-first width search planner; and *Saarplan* (Saar) (Fickert et al. 2018), another portfolio planner.

From the 2023 IPC, we select *Approximate Novelty Search* (ANS) (Singh et al. 2021), which uses approximate best-first width search with novelty approximation and goal count heuristics; *DecStar* (DecS) (Gnad, Torralba, and Shleyfman 2023), based on decoupled search; *Fast Downward Stone Soup 2023* (FDSS23) (Büchner et al. 2023), a portfolio planner; *Maidu* (Corrêa et al. 2023b), a version of the Scorpion planner with width-based search algorithms; and *Levitron* (Corrêa et al. 2023a), a combination of Maidu and the lifted planner Powerlifted.

Year	1998			2000				2002		2004		2006	2008			2011		2014				2018			2023				
	BB2	HSP	IPP	FF	HSP2	MIPS	SysR	LPG	SimP	FD	FDD	MIPSXXL	C3	FFSA	LAMA08	FDSS11	LAMA11	Probe	Merc	MpC	YAHSP	FDRemix	LAPKT	Saar	ANS	DecS	FDSS23	Levitron	Maidu
1998	**7**	**8**	**9**																										
2000	**2**			**29**	**10**	**1**	**10**																						
2002				**23**	**8**		**9**	**2**	**8**																				
2004				**18**	**4**		**9**	**2**	**6**	**7**	**15**																		
2006				**18**	**4**		**9**	**2**	**6**	**7**	**15**																		
2008				**10**	**4**		**8**	**1**	**4**	**5**	**11**		**9**	**4**	**17**														
2011				**8**	**3**		**8**	**1**	**3**	**4**	**6**		**6**	**4**	**8**	**16**	**18**												
2014				**6**	**3**		**8**		**3**	**4**	**5**		**6**	**4**	**5**	**13**	**13**	**9**	**18**	**5**	**7**								
2018				**6**	**3**		**8**		**2**	**4**	**5**		**6**	**4**	**4**	**7**	**9**	**7**	**14**	**5**	**7**	**13**	**10**	**19**					
2023				**5**	**3**		**8**		**2**	**4**	**5**		**6**	**4**	**4**	**6**	**9**	**6**	**13**	**5**	**6**	**12**	**8**	**17**	**11**	**11**	**14**	**14**	**13**

Table 2: Planner First Place Rankings by year. Each cell (year, planner) shows the number of domains where the planner achieves first place in that year, considering only planners available up to that year. Darker shading indicates higher counts. Empty cells indicate zero first places, or that the planner was released in a later year.

Results

We evaluated all planners on our benchmark set using a cluster of Intel Xeon Gold 6130 processors, with a memory limit of 4 GiB and a time limit of 30 minutes.

Table 1 reports the coverage of each planner on a per-IPC basis. The main takeaway is that the highest coverage is achieved by the most recent planners. Another interesting observation from Table 1 is that some earlier planners perform well on domains that were first used in later IPCs. A case in point is Fast Forward, which competed in IPC 2000 yet performs better on domains from IPC 2008 than some more recent planners. Similarly, some later domains are dominated by earlier planners. One example is Scanalyzer, a 2008 domain on which almost no planner—including most from 2023—solves more than 20 instances, whereas System R from 2000 solves all 30. Another example is Satellite, introduced in 2002, where the best coverage is attained by SimPlanner, also from 2002. It solves roughly twice as many instances as recent planners. Still, we can observe that the most recent planners offer the best overall coverage and dominate the largest number of domains.

This trend is more clearly illustrated by Table 2, which counts, for each planner, the number of domains on which it solves the most instances (including ties). The table shows, for instance, that Fast Forward consistently outperforms other planners up to the introduction of LAMA in 2008, which uses the FF heuristic. From that year onward, each competition saw a new planner outperform the previous ones, thereby gradually pushing the coverage frontier forward.

Figure 2 shows the evolution of the maximum coverage achieved by any single planner. It offers a different perspective on the tabular results and confirms that Fast Forward dominates the earlier IPCs. Since 2008, progress has been steady, and each IPC produced planners that outperformed the best planner from the previous competition.

As planners are based on similar approaches and often build upon each other, they sometimes behave similarly. For instance, ANS and LAPKT, which are based on best-first width search, often solve the same instances and fail on the same ones. More generally, Figure 1 groups planners by their per-domain coverage using the Jaccard index, which computes their similarity as follows. Each planner is associated with the set of all instances that it solves, which serves as a fingerprint of its empirical behavior. Planners that solve the same instances have the same fingerprint, and the Jaccard similarity quantifies this with a real number between 0 and 1. If S_A and S_B are the sets of instances solved by planners A and B, respectively, then their similarity is given by $J(A, B) = \frac{|S_A \cap S_B|}{|S_A \cup S_B|}$.

This definition determines how planners are grouped in Figure 1: the dendrogram above the figure iteratively groups the most similar sets of planners together, starting with singletons and repeatedly merging the most similar groups. Note that we only consider the empirical performance of the planners when computing their similarity. Also observe that planners with very low similarity solve very different instances, making them promising candidates for inclusion in a portfolio planner. Additional visualizations are available in the online repository (Lequen et al. 2026a).

Conclusions

In this work, we presented our effort to collect, fix and disseminate distinguished IPC planners. To assess the progress of classical planners over time, we evaluated their coverage on a benchmark set that spans the full history of the IPC. Our results show that, perhaps reassuringly, the most recent planners perform the best. Yet, some older planners exhibit state-of-the-art performance on certain domains.

Future work could provide a finer-grained analysis of these planners and explain the sources of their performance differences across domains. Such an analysis could be complemented by studying planner runtime behavior using other metrics, such as the agile or satisficing score.

Acknowledgments

We are grateful first and foremost to the original authors of all planners considered in this paper. We especially thank the creators of planners from earlier IPCs who provided us with their original implementations after all these years: Stefan Edelkamp for MIPS, Malte Helmert for Fast Diagonally Downward, and Oscar Sapena and Eva Onaindia for SimPlanner. We also thank the organizers of the many editions of the IPC, without whom this study would not have been possible, as well as Malte Helmert for helpful discussions and the anonymous reviewers for their feedback.

This work was supported by the Swedish Research Council under grant number 2024-05403 and by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS), partially funded by the Swedish Research Council through grant agreement no. 2022-06725.

References

2014. *Eighth International Planning Competition (IPC-8): Planner Abstracts*.

2018. *Ninth International Planning Competition (IPC-9): Planner Abstracts*.

2023. *Tenth International Planning Competition (IPC-10): Planner Abstracts*.

Audemard, G.; Paulevé, L.; and Simon, L. 2020. SAT Heritage: A Community-Driven Effort for Archiving, Building and Running More Than Thousand SAT Solvers. In Pulina, L.; and Seidl, M., eds., *Proceedings of the 23rd International Conference on Theory and Applications of Satisfiability Testing (SAT 2020)*, volume 12178 of *Lecture Notes in Computer Science*, 107–113. Springer.

Biere, A.; Fleury, M.; Froleyks, N.; and Heule, M. J. H. 2023. The SAT Museum. In Järvisalo, M.; and Berre, D. L., eds., *Proceedings of the 14th International Workshop on Pragmatics of SAT*, CEUR Workshop Proceedings, 72–87.

Bonet, B.; and Geffner, H. 2001a. Heuristic Search Planner 2.0. *AI Magazine*, 22(3): 77–80.

Bonet, B.; and Geffner, H. 2001b. Planning as Heuristic Search. *Artificial Intelligence*, 129(1): 5–33.

Büchner, C.; Christen, R.; Corrêa, A. B.; Eriksson, S.; Ferber, P.; Seipp, J.; and Sievers, S. 2023. Fast Downward Stone Soup 2023. In (ipc 2023).

Corrêa, A. B.; Francès, G.; Hecher, M.; Longo, D. M.; and Seipp, J. 2023a. Levitron: Combining Ground and Lifted Planning. In (ipc 2023).

Corrêa, A. B.; Francès, G.; Hecher, M.; Longo, D. M.; and Seipp, J. 2023b. Scorpion Maidu: Width Search in the Scorpion Planning System. In (ipc 2023).

Drexler, D. 2024. Loki: A C++20 Library for Parsing and Translating PDDL. https://github.com/drexlerd/Loki.

Dutertre, B. 2020. An Empirical Evaluation of SAT Solvers on Bit-vector Problems. In Bobot, F.; and Weber, T., eds., *Proceedings of the 18th International Workshop on Satisfiability Modulo Theories*, volume 2854 of *CEUR Workshop Proceedings*, 15–25. CEUR-WS.org.

Edelkamp, S. 2001. Planning with Pattern Databases. In Cesta, A.; and Borrajo, D., eds., *Proceedings of the Sixth European Conference on Planning (ECP 2001)*, 84–90. AAAI Press.

Edelkamp, S.; Jabbar, S.; and Nazih, M. 2006. Large-scale optimal PDDL3 planning with MIPS-XXL. *5th International Planning Competition Booklet (IPC-2006)*, 28–30.

Fichte, J. K.; Hecher, M.; and Szeider, S. 2020. A Time Leap Challenge for SAT-Solving. In Simonis, H., ed., *Proceedings of the 26th International Conference on Principles and Practice of Constraint Programming (CP 2020)*, volume 12333 of *Lecture Notes in Computer Science*, 267–285. Springer.

Fickert, M.; Gnad, D.; Speicher, P.; and Hoffmann, J. 2018. SaarPlan: Combining Saarland's Greatest Planning Techniques. In (ipc 2018), 11–16.

Fikes, R. E.; and Nilsson, N. J. 1971. STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*, 2: 189–208.

Geffner, H.; and Bonet, B. 2013. *A Concise Introduction to Models and Methods for Automated Planning*, volume 7 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan & Claypool.

Gerevini, A. E.; Saetti, A.; and Serina, I. 2003. Planning Through Stochastic Local Search and Temporal Action Graphs in LPG. *Journal of Artificial Intelligence Research*, 20: 239–290.

Ghallab, M.; Nau, D.; and Traverso, P. 2004. *Automated Planning: Theory and Practice*. Morgan Kaufmann.

Gnad, D.; Torralba, Á.; and Shleyfman, A. 2023. DecStar-2023. In (ipc 2023).

Goldman, R. P.; Biundo, S.; and Katz, M., eds. 2021. *Proceedings of the Thirty-First International Conference on Automated Planning and Scheduling (ICAPS 2021)*. AAAI Press.

Helmert, M. 2006a. Fast (diagonally) downward. *5th International Planning Competition Booklet*.

Helmert, M. 2006b. The Fast Downward Planning System. *Journal of Artificial Intelligence Research*, 26: 191–246.

Helmert, M.; Röger, G.; Seipp, J.; Karpas, E.; Hoffmann, J.; Keyder, E.; Nissim, R.; Richter, S.; and Westphal, M. 2011. Fast Downward Stone Soup. In *IPC 2011 Planner Abstracts*, 38–45.

Hoffmann, J. 2000. A Heuristic for Domain Independent Planning and its Use in an Enforced Hill-climbing Algorithm. Technical Report 133, University of Freiburg, Department of Computer Science.

Hoffmann, J.; and Nebel, B. 2001. The FF Planning System: Fast Plan Generation Through Heuristic Search. *Journal of Artificial Intelligence Research*, 14: 253–302.

Katz, M.; and Hoffmann, J. 2014. Mercury Planner: Pushing the Limits of Partial Delete Relaxation. In (ipc 2014), 43–47.

Kautz, H.; and Selman, B. 1999. Unifying SAT-based and Graph-based Planning. In Dean, T., ed., *Proceedings of*

the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI 1999), 318–325. Morgan Kaufmann.

Keyder, E.; and Geffner, H. 2008. Heuristics for Planning with Action Costs Revisited. In Ghallab, M.; Spyropoulos, C. D.; Fakotakis, N.; and Avouris, N., eds., *Proceedings of the 18th European Conference on Artificial Intelligence (ECAI 2008)*, 588–592. IOS Press.

Kochemazov, S.; Ignatiev, A.; and Marques-Silva, J. 2021. Assessing Progress in SAT Solvers Through the Lens of Incremental SAT. In Li, C.; and Manyà, F., eds., *Proceedings of the 24th International Conference on Theory and Applications of Satisfiability Testing (SAT 2021)*, volume 12831 of *Lecture Notes in Computer Science*, 280–298. Springer.

Koehler, J.; Nebel, B.; Hoffmann, J.; and Dimopoulos, Y. 1997. Extending Planning Graphs to an ADL Subset. In Steel, S.; and Alami, R., eds., *Recent Advances in AI Planning. 4th European Conference on Planning (ECP 1997)*, volume 1348 of *Lecture Notes in Artificial Intelligence*, 273–285. Springer-Verlag.

Lequen, A.; Joergensen, O.; Phung, W.; Gestrin, E.; Van Meerbeeck, D.; Fritzsche, M.; Drexler, D.; and Seipp, J. 2026a. Planner Museum. https://doi.org/10.5281/zenodo.17861051.

Lequen, A.; Joergensen, O.; Phung, W.; Gestrin, E.; Van Meerbeeck, D.; Fritzsche, M.; Drexler, D.; and Seipp, J. 2026b. Planner Museum. https://github.com/mrlab-ai/planner-museum/tree/icaps-2026.

Lin, F. 2001. A Planner Called R. *AI Magazine*, 22(3): 73–76.

Lipovetzky, N.; Ramirez, M.; and Geffner, H. 2008. C3: Planning with consistent causal chains. *Proceedings International Planning Competition (IPC-6)*, 40: 144.

Lipovetzky, N.; Ramirez, M.; Muise, C.; and Geffner, H. 2014. Width and Inference Based Planners: SIW, BFS(f), and PROBE. In (ipc 2014), 6–7.

McDermott, D.; Ghallab, M.; Howe, A.; Knoblock, C.; Ram, A.; Veloso, M.; Weld, D.; and Wilkins, D. 1998. PDDL – The Planning Domain Definition Language – Version 1.2. Technical Report CVC TR-98-003/DCS TR-1165, Yale Center for Computational Vision and Control, Yale University.

Onaindia, E.; Sapena, Ó.; Sebastia, L.; and Marzal, E. 2001. SimPlanner: An Execution-Monitoring System for Replanning in Dynamic Worlds. In Brazdil, P.; and Jorge, A., eds., *Proceedings of the 10th Portuguese Conference on Artificial Intelligence (EPIA 2001)*, volume 2258 of *Lecture Notes in Computer Science*, 393–400. Springer.

Ramirez, M.; Lipovetzky, N.; and Muise, C. 2015. Lightweight Automated Planning ToolKiT. http://lapkt.org/.

Richter, S.; and Westphal, M. 2008. The LAMA Planner — Using Landmark Counting in Heuristic Search. IPC 2008 short papers, http://ipc.informatik.uni-freiburg.de/Planners.

Richter, S.; Westphal, M.; and Helmert, M. 2011. LAMA 2008 and 2011 (planner abstract). In *IPC 2011 Planner Abstracts*, 50–54.

Rintanen, J. 2014. Madagascar: Scalable Planning with SAT. In (ipc 2014), 66–70.

Seipp, J. 2018. Fast Downward Remix. In (ipc 2018), 74–76.

Seipp, J.; Pommerening, F.; Sievers, S.; and Helmert, M. 2017. Downward Lab. https://doi.org/10.5281/zenodo.790461.

Seipp, J.; Torralba, Á.; and Hoffmann, J. 2022. PDDL Generators. https://doi.org/10.5281/zenodo.6382173.

Singh, A.; Lipovetzky, N.; Ramirez, M.; and Segovia-Aguas, J. 2021. Approximate Novelty Search. In (Goldman, Biundo, and Katz 2021), 349–357.

Singularity Developers. 2021. hpcng/singularity: Singularity 3.7.3.

Taitler, A.; Alford, R.; Espasa, J.; Behnke, G.; Fišer, D.; Gimelfarb, M.; Pommerening, F.; Sanner, S.; Scala, E.; Schreiber, D.; Segovia-Aguas, J.; and Seipp, J. 2024. The 2023 International Planning Competition. *AI Magazine*, 45(2): 280–296.

Torralba, Á.; Seipp, J.; and Sievers, S. 2021. Automatic Instance Generation for Classical Planning. In (Goldman, Biundo, and Katz 2021), 376–384.

Vidal, V. 2014. YAHSP3 and YAHSP3-MT in the 8th International Planning Competition. In (ipc 2014), 64–65.