Unplannability IPC Track

Christian Muise **Nir Lipovetzky**

Department of Computing and Information Systems,
University of Melbourne, Australia
{christian.muise,nir.lipovetzky}@unimelb.edu.au

Abstract

The majority of research in the field of automated planning focuses on the synthesis of plans for problems that are solvable. We propose an IPC track to focus on the important and understudied area of *unplannibility*: proving that a planning problem is unsolvable. We will focus on classical planning problems, as methods for determining whether or not unplannability can have wider applications for classical planning problems (e.g., recognizing and avoiding deadends in the state space) as well as solving planning problems with uncertainty (e.g., identifying when a deterministic approximation of the problem is unsolvable). The unplannability track follows similar contests in other fields; for example, the UNSAT track for the field of Boolean Satisfiability. In a similar vein, we hope that the introduction of an unplannability track will foster new innovation for techniques dedicated to identifying planning problems that cannot be solved.

Motivation

The International Planning Competition (IPC) has had a long history of evaluating the state-of-the-art planners on their ability to synthesize plans for solvable planning problems. However, the need for effective detection of *unsolvable* instances often is overlooked. We propose an IPC track that focuses primarily on the task of *unplannability*: identifying when there is no solution to a given problem.

It is common for competitions in other academic fields to include a combination of both solvable and unsolvable instances in the benchmark domains: see for example the SAT (`http://www.satcompetition.org/`) and constraint programming competitions (`http://www.minizinc.org/challenge.html`). Conversely, the IPC historically has included only solvable instances for the planning benchmarks.[1] In order to focus on and showcase the techniques for detecting unplannability, we feel that an IPC track unique from the satisficing track is warranted.

While largely unstudied, the task of unplannability is an important and difficult problem. As identified by Bäckström et al. (2013), the source of many planning problems naturally have an unsolvable counterpart that is as important, if not more-so, than synthesizing solutions. Examples where it is useful include identifying human error in manually generated plans (Goldman, Kuter, and Schneider 2012), penetration detection of software systems (Boddy et al. 2005), and system verification (Edelkamp, Leue, and Visser 2007) among others. As another example, Hoffmann et al. prove a result for cellular automata by enumerating many initial states for the same goal and identifying which ones have no solution (Hoffmann, Fatès, and Palacios 2010). For this work, they use the sound and complete version of FF (Hoffmann and Nebel 2001).

In the remainder of this proposal, we survey some of the existing techniques that may form a basis for competitors in an unplannability IPC track, outline the specifics of the proposed track such as the timeline and evaluation, and conclude with a discussion.

Existing Approaches

Perhaps the most obvious possible solution to unplannability is any sound and complete planner. This is the approach used by (Hoffmann, Fatès, and Palacios 2010), and coupled with an efficient means of expanding states, this approach can be quite successful. However, more direct methods have been considered in the literature recently that aim to target unplannability in particular.

The first clear attempt to solve problems of unplannability was by Bäckström, Jonsson, and Ståhlberg (2013). In this work, Bäckström et al. construct abstractions of the problem in a systematic way, bounded by a constant k, such that the unplannability of many planning problems can be detected efficiently. Their approach is sound, but incomplete unless k is set to be sufficiently high.

The other approach that aims to tackle unplannability directly, was by Hoffmann, Kissmann, and Torralba (2014). Hoffmann et al. adjust the strategies for merge and shrink so that instead of preserving admissibility, the existence (or lack thereof) of a solution is preserved. This approach extended the capability of unplannability detection presented in (Bäckström, Jonsson, and Ståhlberg 2013), and represents the state of the art in unplannability detection.

Two additional techniques recently were introduced that may be amenable to unplannability detection. Keyder, Hoffmann, and Haslum (2014) introduce an improved mechanism for merging fluents in classical planning problems, and indicate that their approach may assist in the detection

[1] Exceptions to this generalization typically are viewed as "bugs" in the benchmark set.

of deadends (cf. footnote 7 on page 508). Finally, Suda (2014) demonstrates how the new planning technique of property directed reachability can be leveraged to determine unplannability (cf. Section 5.8 on page 306).

The existing techniques offer a variety of options for detecting unplannability, though these are limited in number. An IPC track that focuses on unplannability will help to distinguish which techniques are viable, and pave the way for new methods in detecting unplannability.

Track Details

Time-line

Following similar tracks in the past, the proposed time-line for the unplannability track is as follows:

- **Jun, 2015**: Announcement of the track
- **Jul, 2015**: Call for domains / expression of interest
- **Oct, 2015**: Registration deadline
- **Nov, 2015**: Demo problems provided
- **Jan, 2016**: (optional) Initial feedback on buggy output
- **Feb, 2016**: Domain submission deadline
- **Mar, 2016**: Final planner submission deadline
- **Apr, 2016**: Paper submission deadline
- **Mar - May, 2016**: Contest run
- **Jun, 2016**: Results announced in London

Note the emphasis on a January deadline for the initial planner feedback phase. While any sound and complete planner can be used to check for unplannability, existing software may be prone to errors when run to completion. Consequently, we hope to iterate on the planners submitted to mitigate against submissions containing bugs.

Another important note is that we do not necessarily coincide with the next deterministic IPC track. Because the techniques for satisficing planning and unplannability detection are similar, running the two tracks at alternate times may allow for greater participation.

Scoring

Since there is no standard certificate of unplannability currently, and we do not wish to restrict the type of planners submitted to the inaugural unplannability IPC track, we will focus on the *coverage of problems correctly identified as being unsolvable*. Similar to the optimal IPC track, we will disqualify a solver for a particular domain if it identifies a solvable instance incorrectly as having no solution or vice versa. This increased penalty aims to deter approximate and unsound solutions.

Orthogonal to the coverage score, we will evaluate the speed of each technique following the standard IPC runtime score: sum over all problems T^*/T, where T^* is runtime of the fastest planner. Further, the time score will serve as a means for breaking ties in terms of planner coverage. With regards to computational resources, we will follow the same requirements as the last IPC-8 competition, 6GB RAM and 30 min computational limits.

Benchmark Domains

Bäckström et al. 2013 introduce a set of unsolvable planning instances that was extended subsequently by Hoffmann et al. 2014. We will use these domains with newly generated problem instances as a seed for the domains in the IPC track. Although we cannot guarantee that all (or even most) domains will be ideal unplannability benchmarks, we will strive to find domains with the following properties: (1) the problems are a combination of both satisfiable and unsatisfiable instances; (2) there is no obvious syntactic difference between the satisfiable and unsatisfiable instances; and (3) (at least some of) the satisfiable instances will be difficult for existing satisficing planners to solve. Point (1) is important to avoid submissions of the form "`return True`" and points (2)+(3) are meant to deter submissions of the form "`if FF fails to solve in under k seconds, return True`".

Ultimately, we will aim to use domains that exhibit interesting and useful properties with respect to unplannability and deadend detection; relying in part on the existing problems where deadends and unsolvable instances play a role. Potential sources for existing benchmarks include oversubscribed problems that encode resources, optimal planning benchmarks with the added constraint that a plan should have a cost cheaper than the optimal solution, unsolvable translations of different planning formalisms (e.g., encodings of conformant or contingent problems), etc.

Discussion

We hope that this track will attract attention and foster research on unsolvable tasks, which to date remains an insular topic within the ICAPS community. Our motivation in submitting this proposal to the Workshop on the International Planning Competition is to promote a dialogue for improving the endeavour of an unplannability IPC track. To that end, we have identified the following open questions that we believe are worth discussing with members of the ICAPS community interested in shaping the future IPC tracks:

1. What aspects of unsolvable problems should the benchmarks focus on? E.g., problems where most abstractions fail to recognize unplannibility, or problems with a hidden unplannable core.
2. How do we avoid submissions that "game the system"? The three properties of an ideal benchmark that we listed earlier offer one step towards achieving this. Another is the disqualification of a solver for an incorrect answer. Should the solver source be released for verification? Should a solver be disqualified entirely for an incorrect answer (i.e., false positive or false negative Etc.)?
3. What is an appropriate level of feedback to ensure that all submissions are reasonably free of errors? An initial suite of problems will be constructed to verify that the systems are producing reasonable output for simple problems. Should more than one such iteration exist?
4. Is it reasonable to schedule an unplannability track at a time other than the deterministic track? We argued earlier that this would promote a higher participation in the

unplannability track, but should this track eventually be co-run or even merged with the classical planning IPC? E.g., using a mix of solvable and unsolvable instances in the benchmark domains similar to the satisfiability and constraint programming contests.

Acknowledgements This research is partially funded by the Australian Research Council Discovery and Linkage Grants DP130102825 and LP11010015.

References

Bäckström, C.; Jonsson, P.; and Ståhlberg, S. 2013. Fast detection of unsolvable planning instances using local consistency. In *Proceedings of the Sixth Annual Symposium on Combinatorial Search, SOCS 2013, Leavenworth, Washington, USA, July 11-13, 2013*.

Boddy, M. S.; Gohde, J.; Haigh, T.; and Harp, S. A. 2005. Course of action generation for cyber security using classical planning. In *Proceedings of the Fifteenth International Conference on Automated Planning and Scheduling (ICAPS 2005)*, 12–21.

Edelkamp, S.; Leue, S.; and Visser, W. 2007. Summary of dagstuhl seminar 06172 on directed model checking. In *Directed Model Checking*. Dagstuhl Seminar Proceedings. Dagstuhl, Germany.

Goldman, R. P.; Kuter, U.; and Schneider, T. 2012. Using classical planners for plan verification and counterexample generation. In *AAAI Workshop on Problem Solving Using Classical Planners*.

Hoffmann, J., and Nebel, B. 2001. The FF planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research (JAIR)* 14:253–302.

Hoffmann, J.; Fatès, N.; and Palacios, H. 2010. Brothers in arms? on AI planning and cellular automata. In *ECAI 2010 - 19th European Conference on Artificial Intelligence, Lisbon, Portugal, August 16-20, 2010, Proceedings*, 223–228.

Hoffmann, J.; Kissmann, P.; and Torralba, Á. 2014. "Distance"? Who Cares? Tailoring Merge-and-Shrink Heuristics to Detect Unsolvability. In *ECAI 2014 - 21st European Conference on Artificial Intelligence, 18-22 August 2014, Prague, Czech Republic - Including Prestigious Applications of Intelligent Systems (PAIS 2014)*, 441–446.

Keyder, E. R.; Hoffmann, J.; and Haslum, P. 2014. Improving delete relaxation heuristics through explicitly represented conjunctions. *Journal of Artificial Intelligence Research (JAIR)* 50:487–533.

Suda, M. 2014. Property directed reachability for automated planning. *Journal of Artificial Intelligence Research (JAIR)* 50:265–319.