Towards Thorough Empirical Methods for AI Planning

Jörg Hoffmann* and **Bernhard Nebel**
Institute for Computer Science
<last-name>@informatik.uni-freiburg.de

Abstract

Empirical investigations in the area of AI planning have been focused on a comparatively small set of benchmark tasks. Trying to design larger scale experiments for well-founded empirical reasoning in that area, one encounters a number of severe problems. While some of these problems are inherent to the field, others have plainly been ignored. In our own work, we have made some first steps towards addressing these problems.

The field of domain independent planning is concerned with developing problem solving techniques that are general in the sense that they can—ideally—be used for any application one wants to deal with. Many approaches have been proposed for modeling planning problems, i.e., for formal planning frameworks [Fikes and Nilsson, 1971; Pednault, 1989], as well as for general problem solving strategies within such frameworks [Penberthy and Weld, 1992; Blum and Furst, 1997; Bonet and Geffner, 2001].

For the evaluation of new planning strategies most publications in the area refer to a small set of benchmark planning tasks and make claims on the basis of roughly comparing performance on those tasks. While performance results on a few examples can give a general impression of the usefulness of an approach, a more thorough empirical evaluation is certainly desirable. When we tried to to design and interpret large scale experiments [Hoffmann and Nebel, 2001; Hoffmann, 2001], we encountered a number of problems which roughly fall into the following three categories.

1. Which planning domains should one choose?
2. Which examples should one choose within a domain?
3. What is an adequate formal way of interpreting the experimental data?

As for problem 1, it is inherent in the field, and there does not seem to be much one can do about it. One could, of course, try to generate problem instances completely randomly without any reference to an application domain, similar to what has been done in the area of SAT algorithms [Mitchell *et al.*, 1992]. However, we do not know of any method for generating random planning instances. Furthermore, even if such a method were known, it would be completely unclear whether the generated instances would be representative of the envisioned application.

For these reasons, one usually starts with some planning domains such as *block stacking, logistics*, etc. However, it is not possible to obtain a set of domains representative for all applications one could think of. So one has to choose some domains arbitrarily. In our experiments, we settled for a collection of 20 domains that are justified in the sense that they are (amongst) the most frequently used domains within the planning community.

As for problem 2, this is subdivided into three issues which all have been hardly addressed in the planning literature. Firstly, for almost all benchmark domains there is no definition specifying which planning instances belong to that domain. Secondly, for most domains there are some example instances publicly available, but rarely ever has someone published something about how random instances can be or should be generated. Thirdly, for almost all domains there is no notion of which instances are interesting.

We have dealt with the first point by abstracting from the examples that are publicly available, largely following (yet unpublished) work done by Malte Helmert. Based on that, we have dealt with the second point by choosing and implementing for all our 20 domains the intuitively most obvious randomization strategy. That said, it is clear that we have not (yet) dealt with the third point at all. We have made our random generators publicly available along with descriptions of the randomization strategies. [1] This provides at least a starting

*Georges-Köhler-Allee, Geb. 52, 79110 Freiburg, Germany, Phone: +49 (761) 203-8229, Fax: +49 (761) 203-8222, WWW: http://www.informatik.uni-freiburg.de/ hoffmann

[1] The descriptions and the source code of all generators are available via http://www.informatik.uni-freiburg.de/ hoffmann/ff-domains.html

point into more extensive empirical evaluation.

The problem in interpreting the experimental data stems from the following two facts. Firstly, there are instances within a domain of varying size and what one wants to know is the scaling behavior of an approach. Secondly, in a lot of domains there are by definition only a few instances (sometimes just one instance) per size value.

Given performance results for two planners A and B on a set of examples, we have experimented with a variety of formal ways to judge whether B performs significantly better than A. The parametric statistical procedures we tried were inadequate due to the former point mentioned above. Given a domain with instances of varying size, there simply is no average size or average runtime.

Trying regression analysis failed due to the above second point, there are only few instances per size value—remember that in some domains, there is only a *single* instance per size. Even with many instances, a number of assumptions about the planning method would be necessary, concerning the method's asymptotic runtime behavior in that specific domain. We finally settled for a simple non-parametric test, known as the *two-tailed sign test* [Siegel and N. J. Castellan, 1988]. The test is done by counting the number of times that B performs better than A and deciding about statistical significance via the probability of the observed outcome in a binomial distribution with $p = \frac{1}{2}$ [Hoffmann and Nebel, 2001]. While this approach is not entirely satisfying, it provides at least some formal way of measuring performance in planning.

In summary, we believe that a more thorough treatment of empirical investigation would do good to the planning community. We have made efforts to provide starting points in that direction.

References

[Blum and Furst, 1997] Avrim L. Blum and Merrick L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90(1-2):279–298, 1997.

[Bonet and Geffner, 2001] Blai Bonet and Héctor Geffner. Planning as heuristic search. *Artificial Intelligence*, 2001. Forthcoming.

[Fikes and Nilsson, 1971] Richard E. Fikes and Nils Nilsson. STRIPS: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence*, 2:189–208, 1971.

[Hoffmann and Nebel, 2001] Jörg Hoffmann and Bernhard Nebel. What makes the difference between HSP and FF? In *Proceedings IJCAI-01 Workshop on Empirical Methods in Artificial Intelligence*, 2001.

[Hoffmann, 2001] Jörg Hoffmann. Local search topology in planning benchmarks: An empirical analysis. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, Seattle, Washington, USA, August 2001.

[Mitchell *et al.*, 1992] David Mitchell, Bart Selman, and Hector J. Levesque. Hard and easy distributions of SAT problems. In *Proceedings of the 10th National Conference of the American Association for Artificial Intelligence (AAAI-92)*, pages 459–465, San Jose, CA, July 1992. MIT Press.

[Pednault, 1989] Edwin P.D. Pednault. ADL: Exploring the middle ground between STRIPS and the situation calculus. In R. Brachman, H. J. Levesque, and R. Reiter, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 1st International Conference (KR-89)*, pages 324–331, Toronto, ON, May 1989. Morgan Kaufmann.

[Penberthy and Weld, 1992] J. Scott Penberthy and Daniel S. Weld. UCPOP: A sound, complete, partial order planner for ADL. In B. Nebel, W. Swartout, and C. Rich, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 3rd International Conference (KR-92)*, pages 103–114, Cambridge, MA, October 1992. Morgan Kaufmann.

[Siegel and N. J. Castellan, 1988] S. Siegel and Jr. N. J. Castellan. *Nonparametric Statistics for the Behavioral Sciences*. McGraw-Hill, 2nd edition, 1988.