HOKAGE: A Heuristic Lifted Planning Algorithm for Generating Diverse Generalized Plans

Ugur Kuter and **Robert P. Goldman**
SIFT, LLC
email: {ukuter, rpgoldman}@sift.net

Abstract

This paper describes HOKAGE, a new planning algorithm that (1) performs lifted search for generating solution plans (grounding on the fly during planning and heuristic computations) (2) generates multiple diverse solutions, (3) generates plans with looping structures and patterns in them, and (4) deals with numeric state facts and both numeric and traditional goals. We are currently designing and developing HOKAGE and this paper provides a summary of our ongoing work.

Introduction

Over the years, there have been great strides in automated planning algorithms and planning heuristics that enable those algorithms to find high-quality plans very quickly [*e.g.*.](Gerevini, Saetti, and Serina 2003; Helmert 2006; Hoffmann and Nebel 2001). HOKAGE is a new approach that aims to combine the benefits of both modern heuristic search methods and earlier lifted planning techniques. By doing so, unlike existing heuristic search algorithms, HOKAGE is able to perform arbitrary numeric reasoning during planning and can plan for mixed propositional and numeric goals (i.e., the truck needs to be at a particular location at any time less than 0900 hours). It also generates generalized plans with limited looping structures. Finally, HOKAGE is also able to generate diverse *sets* of generalized plans (Nguyen et al. 2012; Bryce 2014; Roberts, Howe, and Ray 2014; Coman and Muñoz-Avila 2011; Sohrabi et al. 2016), including plan-to-plan diversity measures over looping plan libraries.

The following is a high-level summary of the HOKAGE algorithm: Given a planning domain definition D, a problem specification P, and the number k of diverse solutions requested:

- Construct the first plan based on the relaxed planning graph heuristic, generated via on-the-fly grounding
- Insert the plan into the diverse plan library
- Iterate for k times
  - Randomly select a plan from the plan library
  - Randomly select a point in the plan
  - Keep the prefix of the plan given the selected point, replan the postfix based on the relaxed planning graphs generated via on- the-fly grounding
  - Assess how much the new plan is different than the old one
  - If different by a threshold, then insert the new plan in to the plan library
- Generalize plans based on the procedural structures in them
- Return generalized plans

The subsequent sections provide a summary of the key contributions that we anticipate in the above description.

Lifted heuristic planning

With a planning domain definition and problem as input, which are typically described in a form of PDDL (McDermott 1998; Fox and Long 2003), most state-of-the-art methods first generate a fully-instantiated (ground) variant of those descriptions. This grounding process amalgamates properties of the domain with those of the particular planning problem and thereby create efficient data structures for heuristic search. Although this approach has been shown to be effective and scalable in planning benchmarks, it is very limited in practical applications because of the *a priori* grounding schemes used and the limited expressivity those schemes require (Burstein et al. 2012; Mueller et al. 2017).

On the other hand, several early planners (*e.g.*., SNLP (McAllester and Rosenblitt 1991), SHOP2 (Nau et al. 2003), UCPOP (Penberthy and Weld 1992)) did not require such *a priori* grounding. Although these planners lacked the most recent and successful planning heuristics, they were able to instantiate planning models, control knowledge, and variables on the fly during the course of the planning process. By doing so, such lifted planners avoid the need to enumerate all possible objects or constant symbols – and their combinations – in the problem definition. One of the substantial benefits is that a lifted planner can perform arbitrary numeric reasoning during because it can use the planning state during search as a database as it grounds variables in relevant contexts only.

HOKAGE uses a lifted relaxed planning graph method (an earlier lifted planning graph was in McDermott's

PEDESTAL planner (McDermott 1991)) as a heuristic computation in a best-first search algorithm to generate heuristically-good plans. Starting from the initial state of an input planning problem, HOKAGE uses its lifted relaxed planning graph heuristic to compute heuristic values for all applicable actions in that state.

First HOKAGE creates a state level in the relaxed planning graph from the facts of the initial state. HOKAGE is built on SHOP2's generalized automated theorem prover (Nau et al. 2003),[1] which provides Prolog-like functionality, to identify the planning operators that are applicable in a state and generate ground instantiations of those planning operators as actions for a plan. HOKAGE uses the same theorem-proving expressivity and capabilities as SHOP2. Starting with the initial level, HOKAGE generates subsequent action and state levels of the relaxed planning graph, until the goals of the planning problem are satisfied. Once the relaxed planning graph is unfolded this way, the algorithm uses the same techniques for the heuristic values for every applicable action in the first state level of the graph. The planner then greedily chooses the action that has the highest heuristic value and the search continues with the subsequent state that arises by applying that action in the current state.

A state level in HOKAGE's planning graphs is always grounded. An action-level is created after a state-level by applying the planning operators in that state level: that is, HOKAGE generates most general-unifiers for the preconditions of the action in the particular state level and applies those unifiers to partially-ground the planning operator into an action in the action level. Note that this approach allows for (1) grounding an action on the fly in the context of state level it is applied in and (2) having actions' preconditions contain free variables (*i.e.*, variables that do not appear in the parameters of the action as with the typical STRIPS assumption in PDDL).

Numerical reasoning

A state level can contain predicates with arbitrary numbers and HOKAGE can use them not just as symbols but values in numeric computations during both planning and heuristic computations. For example, the following is a snippet from a state representation in HOKAGE:

```
(vehicle-coords uav1 356536189N 1177322804W)
(vehicle-altitude uav1 5000) ; feet
(have-instrument uav1 eo123)
(scanning-enabled uav1 eo123 5000)
(monitor-eta uav1 1500)
(deadline uav1 1530)
```

Here, the numeric arguments to the logical predicates are not treated as symbolic objects as in some versions of PDDL: instead, HOKAGE recognizes them as numbers and performs reasoning/computation over them, such as geospatial distance measures using the coordinate system or temporal reasoning over the deadlines.

[1]SIFT researchers have generalized this theorem-prover and made it available as an independent library for use outside the SHOP2 planner.

This expressivity allows both the planner and its heuristics to evaluate arbitrary numeric formulas in the preconditions or effects of the action. For example, the following shows a planning operator, using SHOP2's domain definition language, that models how a vehicle slows down given the weather effects:

```
(:op (!slow-down ?vehicle ?orig-eta)
 :precond ((monitor-eta ?vehicle ?orig-eta)
           (weather-impact ?vehicle ?force)
           (assign ?delay (/ ?force 10.0))
           (assign ?new-eta
              (+ ?orig-eta ?delay)))
 :delete ((monitor-eta ?vehicle ?orig-eta))
 :add ((monitor-eta ?vehicle ?new-eta)))
```

This operator computes the delay as a function of the forces exerted by the weather event on the vehicle. It also computes the new estimated time of arrival for the vehicle at its destination based on that delay. With the new information, the planning operator updates the state of the world.

Generating diverse plans

Unlike many existing approaches that are designed to generate the first good solution heuristically very fast, HOKAGE is designed to generate many diverse solutions to an input planning problem. We use a plan-adaptation approach as in the LPG-d diverse planner (Nguyen et al. 2012): once HOKAGE generates the first heuristic plan, it uses local-search to continually adapt it to find new, different plans.

The algorithm first selects a plan from the pool of solutions it has generated so far as the candidate plan for adaptation. Note that since HOKAGE always generates a first heuristically-good solution to the planning problem, there is always an opportunity for diverse planning (if the problem is solvable). Then, the algorithm selects a point in the plan and adapts the postfix of the plan from that point on. In our current implementation, both the candidate plan and the adaptation point are chosen randomly during local search.

Aside from this initial implementation, generating diverse plans, and comparing different approaches to doing so requires a domain-independent, theoretically motivated definition of the diversity (distance) between plans. Previously proposed diversity measures are not theoretically motivated, provide inconsistent results on the same plan sets, and suffer from a number of pathologies. As next steps beyond HOKAGE's current random selections for diverse search, we plan to incorporate our previous work on plan-to-plan distance measure that we developed based on compression theory and Kolmogorov complexity (Goldman and Kuter 2015). In that work, we defined the diversity of plans in terms of how surprising one plan is given another or, its inverse, the conditional information in one plan given another. Kolmogorov complexity provides a domain independent theory of conditional information. While Kolmogorov complexity is not computable, a related metric, Normalized Compression Distance (NCD), provides a well-behaved approximation. In our previous paper, we exhibited a number of pathological cases we have found in existing diversity measures, introduced NCD as an alternative diversity metric, and demonstrated that NCD does not admit those pathologies and that it

provides better diversity information. We plan to investigate how NCD could be used incorporated in the heuristic selection mechanism of diverse planning: in our previous work it was only used as a plan comparison metric.

Plans with looping behavior

Diverse planning has been typically limited to classical plans: *i.e.*, simple sequences of actions. However, in many practical applications plans involve repeating actions or plan segments in which some of the action parameters remain the same and some change, until some criterion is achieved. Such repeating actions or action subsequences constitute looping behavior. When plans do not capture that structure, they can be excessively long, and are not efficiently constructed by existing planners. In addition, existing measures of plan diversity do not take repeated structure into account in their assessments of how different plans are.

For example, the following is an example generated by the planner in our weather scenario:

```
((!CLOUD-COVER-FORMED CLOUD-COVER
  ;; Latitude and longitude
  356536189N 1177322804W
  3000) ; Altitude
 (:LOOP
   (:FOR ?LOOPVAR120995)
   (:FROM 1500) ;; time
   (:TO 1515.0)
   (:DO
    ((!SLOW-DOWN UAV-1 ?LOOPVAR120995)))))
```

The above plan shows a weather event with cloud cover forming at a particular location and causing delay to a UAV in that location, over a certain time. A classical plan represents this plan with a series of SLOW-DOWN actions whereas it is really a looping behavior.

Our work here is similar to Srivastava,*et al.*. (2008), however, HOKAGE like most planners, aims to solve individual planning problems, rather than generalize plans to cover multiple situations. Our current implementation is a first step in this approach. HOKAGE currently traverses a plan to recognize repeated patterns of actions. Once the repeated patterns are found, HOKAGE identifies those parameters of the actions that change value. These are posted as candidate loop variables (*e.g.*., `?LOOPVAR120995` in the above example). The grounded values in the plan actions constitute the bounds for the loop; if the hypothesized loop variable is a numeric one, HOKAGE generates the start and the end values for that variable as seen in the plan. Otherwise, the possible value domain for the loop variable is represented as a set.

From a diverse planning perspective, different timelines that could be exhibited by these kinds of plans could generate different lengths of actions. In our opinion, that should not be seen as increasing the diversity of the plans because in all of those cases, the looping behavior is essentially the same. Our intent is for HOKAGE to take the generalized plan structures into account during diverse planning semantically and theoretically.

Conclusions and Ongoing Work

Our work with the HOKAGE system is just beginning. HOKAGE is fully implemented, but we continue to work to improve it. We are also working to provide a formal characterization of the class of problems that HOKAGE can solve. One of the interesting questions in HOKAGE is how to formalize and reason with secondary preconditions and conditional effects.

After this process has been completed, we will begin conducting experiments to probe HOKAGE's efficiency, comparing it to state-of-the-art planners such as LPG-d (Nguyen et al. 2012), on benchmark domains and problems relevant to HOKAGE's capabilities.

Acknowledgments. This work was supported in part by the AFRL Information Directorate through contract FA8750-16-C-0148. Approved for Public Release, Distribution Unlimited. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

References

Bryce, D. 2014. Landmark-based plan distance measures for diverse planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*.

Burstein, M.; Goldman, R.; Robertson, P.; Laddaga, R.; Balzer, R.; Goldman, N.; Geib, C.; Kuter, U.; McDonald, D.; Maraist, J.; Keller, P.; and Wile, D. 2012. Stratus: Strategic and tactical resiliency against threats to ubiquitous systems. In *Proceedings of SASO-12*.

Coman, A., and Muñoz-Avila, H. 2011. Generating diverse plans using quantitative and qualitative plan distance metrics. In *Proceedings AAAI*.

Fox, M., and Long, D. 2003. PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*.

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

Goldman, R. P., and Kuter, U. 2015. Measuring plan diversity: Pathologies in existing approaches and a new plan distance metric. In *Proceedings of AAAI Conference*. Menlo Park, CA: Proc. National Conf. on Artificial Intelligence (AAAI).

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

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

McAllester, D., and Rosenblitt, D. 1991. Systematic nonlinear planning. In *Proceedings of the Ninth National Conference on Artificial Intelligence*, 634–639. Cambridge, MA: MIT Press.

McDermott, D. V. 1991. Regression planning. *International Journal of Intelligent Systems* 6(4):357–416.

McDermott, D. 1998. PDDL, the planning domain definition language. Technical report, Yale Center for Computational Vision and Control.

Mueller, J. B.; Miller, C. A.; Kuter, U.; Rye, J.; and Hamell, J. 2017. A human-system interface with contingency planning for collaborative operations of unmanned aerial vehicles. In *AIAA Information Systems-AIAA Infotech@ Aerospace (2017-1296)*. AIAA Press.

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Murdock, W.; Wu, D.; and Yaman, F. 2003. SHOP2: An HTN planning system. *JAIR* 20:379–404.

Nguyen, T. A.; Do, M. B.; Gerevini, A.; Serina, I.; Srivastava, B.; and Kambhampati, S. 2012. Generating diverse plans to handle unknown and partially known user preferences. *Artificial Intelligence* 190:1–31.

Penberthy, J. S., and Weld, D. S. 1992. UCPOP: a sound, complete, partial order planner for ADL. In Nebel, B.; Rich, C.; and Swartout, W., eds., *Principles of Knowledge Representation and Reasoning:Proceedings of the Third International Conference*, 103–114. Waltham, MA: Morgan Kaufmann.

Roberts, M.; Howe, A.; and Ray, I. 2014. Evaluating diversity in classical planning. In *Proceedings ICAPS*.

Sohrabi, S.; Riabov, A.; Udrea, O.; and Hassanzadeh, O. 2016. Finding diverse high-quality plans for hypothesis generation. In *Proceedings of the 22nd European Conference on Artificial Intelligence (ECAI)*.

Srivastava, S.; Immerman, N.; and Zilberstein, S. 2008. Learning generalized plans using abstract counting. In Fox, D., and Gomes, C. P., eds., *Proceedings AAAI*, 991–997. AAAI Press.