Combining Heuristic Search with Hierarchical Task-Network Planning: A Preliminary Report*

Nathaniel Waisbrot and **Ugur Kuter**
University of Maryland,
Department of Computer Science and
Institute for Advanced Computer Studies
College Park, Maryland 20742, USA
{waisbrot,ukuter}@cs.umd.edu

Tolga Könik
Stanford University and
Institute for the Study of Learning and Expertise
2164 Staunton Court
Palo Alto, California 94306, USA
konik@stanford.edu

Abstract

Important advances in automated planning have been made recently, especially with the development of *domain-configurable* planning systems. These planners use a domain-independent search engine for planning, but they have also the ability to exploit domain-specific planning knowledge. Examples of such planners include the well-known TLPLAN (Bacchus & Kabanza 2000), TALPLANNER (Kvarnström & Doherty 2001), and SHOP2 (Nau *et al.* 2003).

One challenge for domain-configurable planners is that they require a domain expert to provide planning knowledge to the system. When this knowledge is not accurate, complete, poorly expressed, the performance of these planners diminishes considerably and very quickly, even in simple planning benchmarks. In this paper, we present a preliminary report on our research aimed to mitigate this issue by combining the use of domain-specific knowledge and domain-independent heuristic search. We describe H_2O (short for *Hierarchical Heuristic Ordered planner*), a new *Hierarchical Task-Network (HTN)* planning algorithm that can heuristically select the best task decompositions by using domain-independent state-based heuristics.

Our experiments in the DARPA Transfer Learning Program demonstrated the potentialities of H_2O: given HTNs generated by a machine-learning system, which were much less optimal than an expert would encode, H_2O was able to solve problems that SHOP2 could not.

H_2O: HTN Planning with Heuristic Search

For modeling structured domain knowledge, one of the best-known approaches is HTN planning. An HTN planner formulates a plan by decomposing tasks (i.e., symbolic representations of activities to be performed) into smaller and smaller subtasks until primitive actions are reached. The basic idea was developed by (Sacerdoti 1975; Tate 1977), and the formal underpinnings were developed in (Erol, Hendler, & Nau 1996). The more recent SHOP2 planner (Nau *et al.* 2003) has been very successful both in solving planning benchmarks (Fox & Long 2002) and as a deployed system in many real-world applications (Nau *et al.* 2005). The primary difference between SHOP2 and most other HTN planners is that SHOP2 plans for tasks in the same order that they will be executed, and thus it knows the current state at each step of the planning process. This reduces the complexity of reasoning by removing a great deal of uncertainty about the world and allows for substantial expressive power.

In this paper, we used the same definitions of states, primitive and nonprimitive tasks, planning operators, actions, inference axioms, plans, and HTN planning problems as in SHOP2 (Nau *et al.* 2003). We extended the definition of an HTN method (i.e., an operational procedure that describes how nonprimitive tasks are decomposed into their subtasks) to include a *goal expression*, i.e., a single logical atom that will be true in the state of the world when/if all of the subtasks are successfully accomplished in the current state.[1] A method's goal expression describes the possible goal states reachable by the planner from the current state by accomplishing the current task. This enables us to use domain-independent heuristics during task decomposition.

H_2O starts with an initial set T of high-level tasks. At each iteration, the algorithm chooses a task t from T that does not have any predecessors. If t is a primitive task, then H_2O generates an action for it. If t is not primitive, then the planner selects an HTN method for t, decomposes t into its subtasks using that method, and inserts the subtasks into T while ensuring the ordering and variable-binding constraints that are imposed by the method are met correctly in the updated T. Then, the planner recursively calls itself to process any other nonprimitive task in T in the same way as above. This recursive process continues until all nonprimitive tasks are decomposed into primitive tasks (i.e., actions). At that point, the primitive task network corresponds to a solution plan and H_2O returns it.

An important difference between H_2O and SHOP2 is the way H_2O selects an HTN method for a task t. SHOP2 chooses one of the alternative methods based on the order they are specified in the input, applies that method to generate subtasks, and backtracks in the case of failure. In H_2O, this is a point of *heuristic selection*: H_2O does not re-

*This work was supported in part by DARPA's Transfer Learning and Integrated Learning programs. The opinions in this paper are those of the authors and do not necessarily reflect the opinions of the funders.

[1]The use of goal expressions associated with HTN methods is originated from the previous work on ICARUS, a machine-learning system capable of producing hierarchical knowledge similar to HTNs (Nejati, Langley, & Könik 2006).

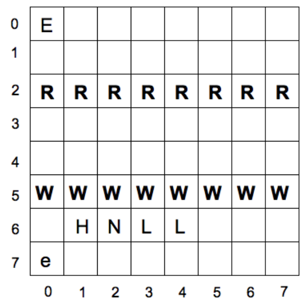

Figure 1: One of the Escape problems used in the Transfer Learning program. The explorer agent, e, first need to collect the hammer, nails, and logs (H, N, L). It then combines the nails and the logs to build a bridge across the water (W), and then breaks the rocks (R) to clear a path to the exit, E.

quire the expert-encoded method ordering as input; instead, it heuristically selects the best applicable method for t.

H_2O takes as input a domain-independent heuristic function and uses it computes how close the goal of the method is to the state in which the decomposition is performed. Given the current state and the goal expression of a method, the heuristic function computes a score value of using that method for the current task in the current state. The planner can compute a score using any heuristic function originally developed for existing state-space search planners, such as *Manhattan Distance* heuristic and the distance-based heuristics as in FF (Hoffmann & Nebel 2001).

Implementation and Experiments

We have used H_2O in the DARPA Transfer Learning (TL) program. One of the benchmark problems used in the evaluations is illustrated in Figure 1, where the objective is to explore across a $n \times m$ grid to reach the exit square, picking up and constructing tools to cross barriers along the way. For further details on the TL benchmarks and H_2O's performance on them, see (Waisbrot 2007).

In the experiments, the HTN methods and our axioms were produced by the ICARUS machine-learning system (Nejati, Langley, & Könik 2006). For example, the top-level tasks learned by ICARUS in the above scenario were ((HOLDING TOOLS) (COMPROMISED ?WATER) (DESTROYED ?ROCKS) (ATEXIT)).[2] Here, picking up the nails using the method (HOLDING NAILS) would decompose into a task for taking the agent to (LOCATION 2 6), but without any understanding of how close this was to the agent, or in which direction it was, the planner plots a sub-optimal course around the grid to accomplish that task.

When run with SHOP2, the available HTNs exhausted its stack space before solving the majority of the problems since they induced frequent backtracking and had a large branching factor. H_2O, on the other hand, was able to alleviate this problem by using a simple Manhattan Distance heuristic. For each goal that involved moving to a grid location, the heuristic directed decomposition to follow the shortest path. With the machine-generated HTNs providing high-level control of tasks and the Manhattan distance heuristic providing low-level control, H_2O solved the benchmark problems rapidly (Waisbrot 2007). In the scenario of Figure 1, for example, H_2O was able to generate a solution in 6.7 seconds, whereas SHOP2, which cannot exploit any heuristics, failed after nearly 20 minutes.

[2]ICARUS annotates its output methods with the main effect that each of the method was intended to produce. We used these annotations as the goal expressions of the methods.

Conclusions

We have described H_2O, a planning algorithm that combines heuristic search with HTN planning. For each task to be decomposed, the planner heuristically selects the best decomposition among the possible ones induced by the set of applicable HTN methods. This combination is particularly promising when the HTNs are inaccurate, incomplete, or poorly expressed, produced by either a machine learning system or a human who is not an expert in the particular domain or at HTN writing.

We are currently developing a general theory of using heuristic search in the context of HTN planning and performing an extensive evaluation of the approach.

References

Bacchus, F., and Kabanza, F. 2000. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence* 116(1-2):123–191.

Erol, K.; Hendler, J.; and Nau, D. S. 1996. Complexity results for hierarchical task-network planning. *AMAI* 18:69–93.

Fox, M., and Long, D. 2002. International planning competition. http://planning.cis.strath.ac.uk/competition.

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

Kvarnström, J., and Doherty, P. 2001. TALplanner: A temporal logic based forward chaining planner. *Annals of Mathematics and Articial Intelligence* 30:119–169.

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

Nau, D.; Au, T.-C.; Ilghami, O.; Kuter, U.; Muñoz-Avila, H.; Murdock, J. W.; Wu, D.; and Yaman, F. 2005. Applications of SHOP and SHOP2. *IEEE Intelligent Systems* 20(2):34–41.

Nejati, N.; Langley, P.; and Könik, T. 2006. Learning hierarchical task networks by observation. In *ICML*.

Sacerdoti, E. 1975. The nonlinear nature of plans. In *IJCAI*.

Tate, A. 1977. Generating project networks. In *IJCAI*.

Waisbrot, N. 2007. A Description of the Evaluation Benchmarks in the Year 2 of the DARPA Transfer Learning Program. http://www.cs.umd.edu/users/waisbrot/darpa-tl/.