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

**Author 1**
Affliation of the Author 1
Email of the Author 1

**Author 2**
Affliation of the Author 2
Email of the Author 2

**Author 3**
Affliation of the Author 3
Email of the Author 3

**Abstract**

Some of the important advances in automated Artificial Intelligence Planning in recent years have been the development of *domain-configurable* planning systems. These planners are built on top of domain-independent search engines but they have also the ability to exploit domain-specific planning knowledge during planning. Examples of such planning systems 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 give 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 to compute how close the goal achieved by a decomposition is to the current state. We used $H_2O$ recently in the evaluations of the DARPA Transfer Learning Program (Oblinger 2005). The results demonstrated the potentialities of our approach: given HTNs generated by a machine-learning system, which were less than optimal compared to ones that an expert would encode, $H_2O$ using the simple *Manhattan Distance* heuristic was able to outperform SHOP2. We discuss these results with a representative example from our tests.

## $H_2O$: 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). Recently, the SHOP2 planner has been demonstrated to be very successful both in the International Planning Competition (Fox & Long 2002) and as a deployed system in many real-world applications (Nau *et al.* 2005).

In this paper, we used the same definitions of states, primitive and nonprimitive tasks, HTN methods (i.e., operational procedures that describe how nonprimitive tasks are decomposed into their subtasks), planning operators, actions, inference axioms, plans, and HTN planning problems as in SHOP2 (Nau *et al.* 2003). We extended the definition of a method 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. 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 existing HTN planners such as SHOP2, to the best of our knowledge, is the way $H_2O$ selects an HTN method for a task. For example in SHOP2, this is a nondeterministic choice; the planner chooses one of the alternative methods, applies it to generate subtasks, and backtracks in the case of failure. In $H_2O$, this is a point of *heuristic selection*: the algorithm heuristically selects the best method that is applicable to the current nonprimitive task $t$. To accomplish this, $H_2O$ uses the goal expressions of the HTN methods as described above and the heuristic function computes how close these goals are to the state in which the decomposition is performed.

More specifically, 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. Then, $H_2O$ selects the method with the best score. In the absence of good HTN methods that encode expert search-control strategies, this approach provides a way to compensate the available HTN methods. The planner can

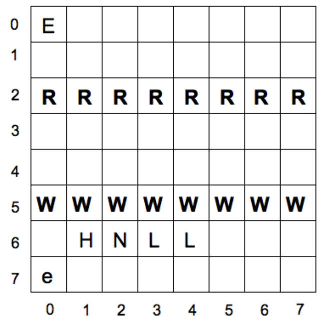


Figure 1: One of the Escape problems used in the Transfer Learning program. The explorer agent, `e`, first need to collect the hammer, nails, and logs (`H`, `N`, `L`). It then combines the nails and the logs to build a bridge across the water (`W`), and then breaks the rocks (`R`) to clear a path to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

use any heuristic function originally developed for existing state-space search planners, such as the *Manhattan Distance* heuristic and the distance-based heuristics as in FF (Hoffmann & Nebel 2001).

## Implementation and Experiments

One benchmark in the DARPA Transfer Learning (TL) program recently was the Escape domain, where an agent moves across a $n \times m$ grid to reach the exit square, picking up and constructing tools to cross barriers along the way.

The HTN methods and our axioms were produced by the ICARUS machine-learning system (Nejati, Langley, & Könik 2006).[1] 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. For example, one of the benchmark problems used in the evaluations is illustrated in Figure 1, where the top-level tasks were ((HOLDING TOOLS) (COMPROMISED ?WATER) (DESTROYED ?ROCKS) (ATEXIT)). Here, picking up the nails using the method (HOLDING NAILS) would decompose into a task for taking the agent to (LOCATION 3 6), but without any understanding of how close this was to the agent, or in which direction it was, the planner plots a suboptimal course around the grid to accomplish that task.

$H_2O$, 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 Escape problem set rapidly. 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, exhausted the stack space and failed after nearly 20 minutes with the given HTNs.

[1] 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.

$H_2O$ produced similar results on two other, more complex domains in the TL program, but we defer the discussion of those results to another paper due to space limitations.

## Conclusions

We have described a new planning system that is capable of generating plans via heuristic search and hierarchical task decomposition. 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. Thus, the planner combines the power of planning via domain-independent heuristic selections and HTN planning. 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 domain-independent heuristic search in the context of HTN planning. This work will also include an extensive experimental evaluation of the approach, using several planning benchmarks from the previous International Planning Competitions (Fox & Long 2002; Edelkamp & Hoffmann 2004).

## References

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

Edelkamp, S., and Hoffmann, J. 2004. International planning competition. http://ipc.icaps-conference.org.

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*.

Oblinger, D. 2005. Darpa transfer learning program. http://www.darpa.mil/ipto/solicitations/closed/05-29_PIP.htm.

Sacerdoti, E. 1975. The nonlinear nature of plans. In *IJCAI*, 206–214.

Tate, A. 1977. Generating project networks. In *IJCAI*, 888–893.