Tight Bounds for HTN Planning with Task Insertion (Extended Abstract)*

Ron Alford
ASEE/NRL Postdoctoral Fellow
Washington, DC, USA
ronald.alford.ctr@nrl.navy.mil

Pascal Bercher
Ulm University
Ulm, Germany
pascal.bercher@uni-ulm.de

David W. Aha
U.S. Naval Research Laboratory
Washington, DC, USA
david.aha@nrl.navy.mil

Abstract

Hierarchical Task Network (HTN) planning with task insertion (TIHTN planning) is a variant of HTN planning. In HTN planning, the only means to alter task networks is to decompose compound tasks. In TIHTN planning, tasks may also be inserted directly. In this paper we provide tight complexity bounds for TIHTN planning along two axis: whether variables are allowed and whether methods must be totally ordered.

Background & Motivation

The Hierarchical Task Network (HTN) planning paradigm is centered around planning the execution of tasks (Erol, Hendler, and Nau 1996). Fundamental to HTN planning is the *task network*, which is a partially-ordered multiset of task names, each representing an activity to be accomplished. Tasks are either primitive, corresponding to an action that can be taken whenever its precondition is met, or non-primitive, which must be iteratively refined into an executable sequence of actions. There are two types of refinement available to an HTN planner: either imposing an ordering on two tasks, restricting the task network's partial order, or by *decomposing* a non-primitive task. For every non-primitive task name, there is an associated set of methods, each containing a task network. We decompose a non-primitive task name by replacing it with the task network from one of its associated methods.

Although HTN planning has found numerous practical applications (Nau et al. 2005), it can be difficult to write complete and effective sets of methods (Shivashankar et al. 2013). A principled approach to this problem is that of HTN planning with Task Insertion (TIHTN Planning), which adds an additional refinement operator beyond that of the HTN formalism: the ability to insert any task name into the task network without regard to the decomposition hierarchy (Geier and Bercher 2011). Task insertion is also allowed by other hierarchical planning approaches, such as *hybrid planning* (Kambhampati, Mali, and Srivastava 1998; Biundo and Schattenberg 2001) – a planning approach fusing HTN planning with Partial-Order Causal-Link (POCL) planning. Task insertion can also be seen as the underlying formalism in hierarchical goal systems such as GoDel (Shivashankar et al. 2013).

Somewhat surprisingly, allowing task insertion reduces the worst case complexity of deciding the plan existence problem. While even propositional HTN planning is undecidable, propositional TIHTN planning is known to be in EXPSPACE (Geier and Bercher 2011). Here, we lower that bound for the general case and further investigate the influence of variables (or lifting) and the ordering of the task networks given in the decomposition methods. We give a colloquial overview of our results. For formal definitions and proofs we refer to the full paper (Alford, Bercher, and Aha 2015b).

Results and Comparison

One of the key insights into TIHTN planning is that any solution can be mimicked by a mixture of task insertion and *acyclic decomposition*, where no task name is ever decomposed using a method that contains the same task name as an ancestor (Geier and Bercher 2011). We can exploit this insight to define *acyclic progression* for TIHTN problems using the progression planning technique commonly used in HTN planning (Alford et al. 2012).

Under acyclic progression, a TIHTN planning algorithm needs only to consider a small portion of the task network at the time. Using that technique we obtain tighter bounds for TIHTN planning given only totally-ordered methods.

Theorem 1. *TIHTN planning for propositional problems with totally-ordered methods is* PSPACE*-complete. For lifted problems (having variables) with totally-ordered methods, TIHTN planning is* EXPSPACE*-complete.*

For partially-ordered TIHTN problems, acyclic progression may need decompose all or most of the task network before imposing ordering constraints. The number of required task insertions, however, is limited by the number of expressible states.

Theorem 2. *Propositional TIHTN planning is* NEXPTIME*-complete; lifted TIHTN planning is 2-*NEXPTIME*-complete.*

Partially-ordered problems are not always more computationally challenging than totally-ordered problems. *Regular* problems are those where every method contains at most

*Full version will appear in IJCAI-2015

Vars.	Ordering	TI	Recursion	Complexity
no	total	no	acyclic	PSPACE
no	total	no	regular	PSPACE
no	total	no	tail	PSPACE
no	total	no	arbitrary	EXPTIME
no	total	yes	–	PSPACE
no	partial	no	acyclic	NEXPTIME
no	partial	no	regular	PSPACE
no	partial	no	tail	EXPSPACE
no	partial	no	arbitrary	undecidable
no	partial	yes	regular	PSPACE
no	partial	yes	–	NEXPTIME
yes	total	no	acyclic	EXPSPACE
yes	total	no	regular	EXPSPACE
yes	total	no	tail	EXPSPACE
yes	total	no	arbitrary	2-EXPTIME
yes	total	yes	–	EXPSPACE
yes	partial	no	acyclic	2-NEXPTIME
yes	partial	no	regular	EXPSPACE
yes	partial	no	tail	2-EXPSPACE
yes	partial	no	arbitrary	undecidable
yes	partial	yes	regular	EXPSPACE
yes	partial	yes	–	2-NEXPTIME

Table 1: A comparison of the complexity classes for HTN planning (Alford, Bercher, and Aha 2015a) with TIHTN planning (Alford, Bercher, and Aha 2015b). The TIHTN planning results are indicated by TI=yes. All results are completeness results.

one non-primitive task, and that task is constrained to come after all others (Erol, Hendler, and Nau 1996). For regular problems, neither subtask ordering nor task insertion (or lack thereof) affect worst-case asymptotic bounds:

Corollary 3. *TIHTN plan-existence for regular problems is* PSPACE*-complete when they are propositional, and* EXPSPACE*-complete for lifted problems, regardless of ordering.*

Discussion

Our major contribution is in providing six new completeness results for common classes of TIHTN planning, including better bounds for the general case. This provides both worst-case lower bounds for any sound and complete TIHTN algorithm; and matching worst-case upper bounds, which provides a theoretical target for future algorithms to match. The lower bounds, in particular, should help focus future research, as they provide firm limits on what sound and complete algorithms exist, e.g., there is no polynomial translation of all TIHTN problems into classical planning unless PSPACE = NEXPTIME.

In our proofs, the acyclic progression technique was instrumental in proving upper bounds that matched the lower bounds for each of the problem classes we examined. As such, it is at least a baseline algorithm to compare against in future work, likely in the context of modern hybrid planners such as PANDA (Bercher, Keen, and Biundo 2014), or the more loosely hierarchical ANML (Smith, Frank, and Cushing 2008).

Acknowledgment

This work is sponsored in part by OSD ASD (R&E) and by the Transregional Collaborative Research Centre SFB/TRR 62 “Companion-Technology for Cognitive Technical Systems” funded by the German Research Foundation (DFG). The information in this paper does not necessarily reflect the position or policy of the sponsors, and no official endorsement should be inferred.

References

Alford, R.; Bercher, P.; and Aha, D. W. 2015a. Tight bounds for HTN planning. In *Proc. of the 25th Int. Conf. on Automated Planning and Scheduling (ICAPS)*. AAAI Press.

Alford, R.; Bercher, P.; and Aha, D. W. 2015b. Tight bounds for HTN planning with task insertion. In *Proc. of the 25th Int. Joint Conf. on Artificial Intelligence (IJCAI)*. AAAI Press.

Alford, R.; Shivashankar, V.; Kuter, U.; and Nau, D. S. 2012. HTN problem spaces: Structure, algorithms, termination. In *Proc. of the 5th Annual Symposium on Combinatorial Search (SoCS)*, 2–9. AAAI Press.

Bercher, P.; Keen, S.; and Biundo, S. 2014. Hybrid planning heuristics based on task decomposition graphs. In *Proc. of the Seventh Annual Symposium on Combinatorial Search (SoCS)*, 35–43. AAAI Press.

Biundo, S., and Schattenberg, B. 2001. From abstract crisis to concrete relief (a preliminary report on combining state abstraction and HTN planning). In *Proc. of the 6th Europ. Conf. on Planning (ECP)*, 157–168. AAAI Press.

Erol, K.; Hendler, J. A.; and Nau, D. S. 1996. Complexity results for HTN planning. *Annals of Mathematics and Artificial Intelligence* 18(1):69–93.

Geier, T., and Bercher, P. 2011. On the decidability of HTN planning with task insertion. In *Proc. of the 22nd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 1955–1961. AAAI Press.

Kambhampati, S.; Mali, A.; and Srivastava, B. 1998. Hybrid planning for partially hierarchical domains. In *Proc. of the 15th Nat. Conf. on Artificial Intelligence (AAAI)*, 882–888. AAAI Press.

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

Shivashankar, V.; Alford, R.; Kuter, U.; and Nau, D. 2013. The GoDeL planning system: a more perfect union of domain-independent and hierarchical planning. In *Proc. of the 23rd Int. Joint Conf. on Artificial Intelligence (IJCAI)*, 2380–2386. AAAI Press.

Smith, D. E.; Frank, J.; and Cushing, W. 2008. The ANML language. In *Proc. of the Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.