sfb transregio 62
Companion Technology

# On the Decidability of HTN Planning with Task Insertion

**Thomas Geier and Pascal Bercher**

Institute of Artificial Intelligence, Ulm University, Germany

IJCAI 2011, International Joint Conference on Artificial Intelligence, 19-22 July 2011, Barcelona, Spain


## Abstract

We give a simplified propositional HTN formalization inspired by the formalization of Erol et al. [2] and show

- the plan existence problem is still undecidable despite the simplifications
- HTN planning with insertion (hybrid planning) is decidable; from the proof of decidability, we obtain an upper complexity bound of $\mathbf{EXPSPACE}$ for the plan existence problem for propositional hybrid planning


## Definition (Task Network)

A *task network* $\mathrm{tn} = (T, \prec)$ is a partially ordered sequence of tasks:

- $T$ is a finite and non-empty set of tasks
- $\prec \subseteq T \times T$ is a strict partial order on $T$

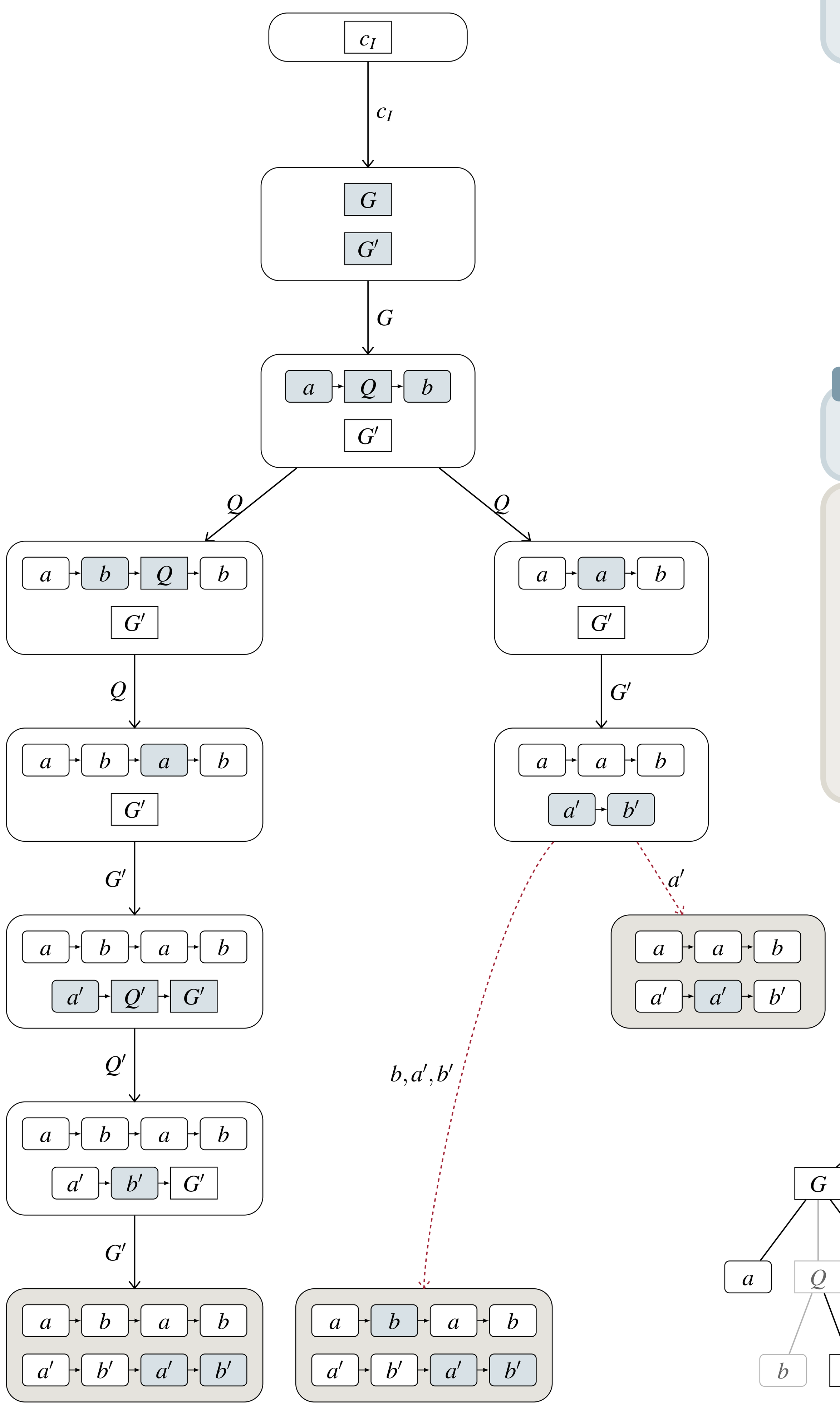
## Definition (Planning Problem)

A *planning problem* is a 6-tuple $P = (V, C, O, M, c_I, s_I)$ and

- $V$ is a finite set of *state variables*
- $C$ is a finite set of *compound tasks*
- $O$ is a finite set of *primitive tasks*, for $o \in O$, $(\mathrm{prec}(o), \mathrm{add}(o), \mathrm{del}(o)) \in 2^V \times 2^V \times 2^V$ is an operator
- $M \subseteq C \times TN$ is a finite set of *decomposition methods*
- $c_I \in C$ is the *initial task*
- $s_I \in 2^V$ is the *initial state*

Note that the part of the planning problem that is usually called the *domain* (tasks and methods) is given with the problem.

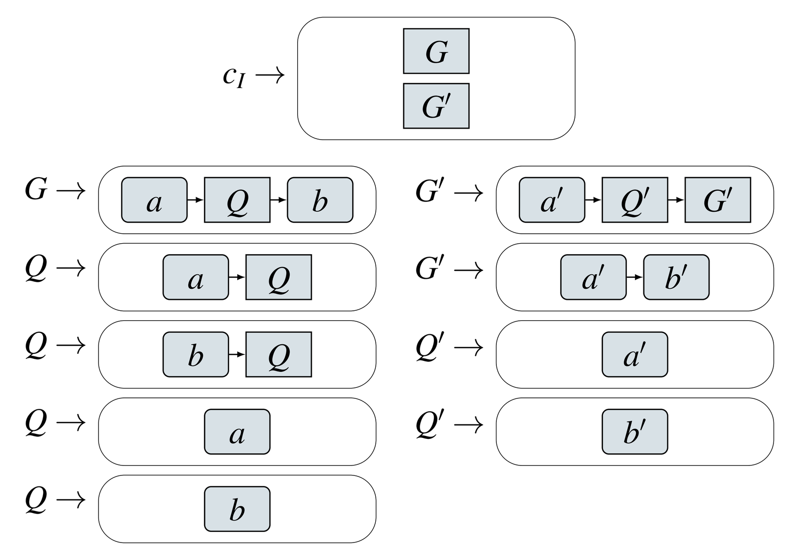
**Figure 1:** *a search space fragment*



## Possible Modifications

1. Decomposition:
   - given a task network $\mathrm{tn} = (T, \prec)$, use method $(t, \mathrm{tn}') \in M$ to replace $t \in T$ by $\mathrm{tn}'$. Then adjust ordering constraints.
2. Insertion:
   - insert primitive tasks from $O$
   - insert ordering constraints

**Figure 2:** *decomposition methods*

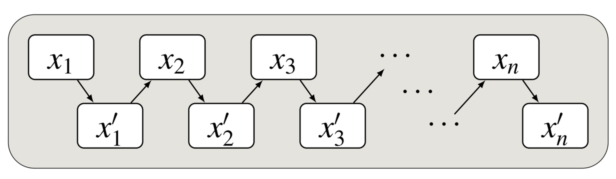


Here, the decomposition methods describe two context-free grammars (CFGs); their languages are $L(G) = a(a|b)^+b$ and $L(G') = (a'(a'|b'))^*a'b'$.

## Solution Criterion

1. A task network $\mathrm{tn}$ is an HTN solution iff:
   - $\mathrm{tn}$ is obtained via decomposition
   - there is an executable linearization of $\mathrm{tn}$'s tasks
2. A task network $\mathrm{tn}$ is a hybrid solution iff:
   - $\mathrm{tn}$ is obtained by decomposition *followed by insertion*
   - there is an executable linearization of $\mathrm{tn}$'s tasks

**Figure 3:** *structure of solutions*



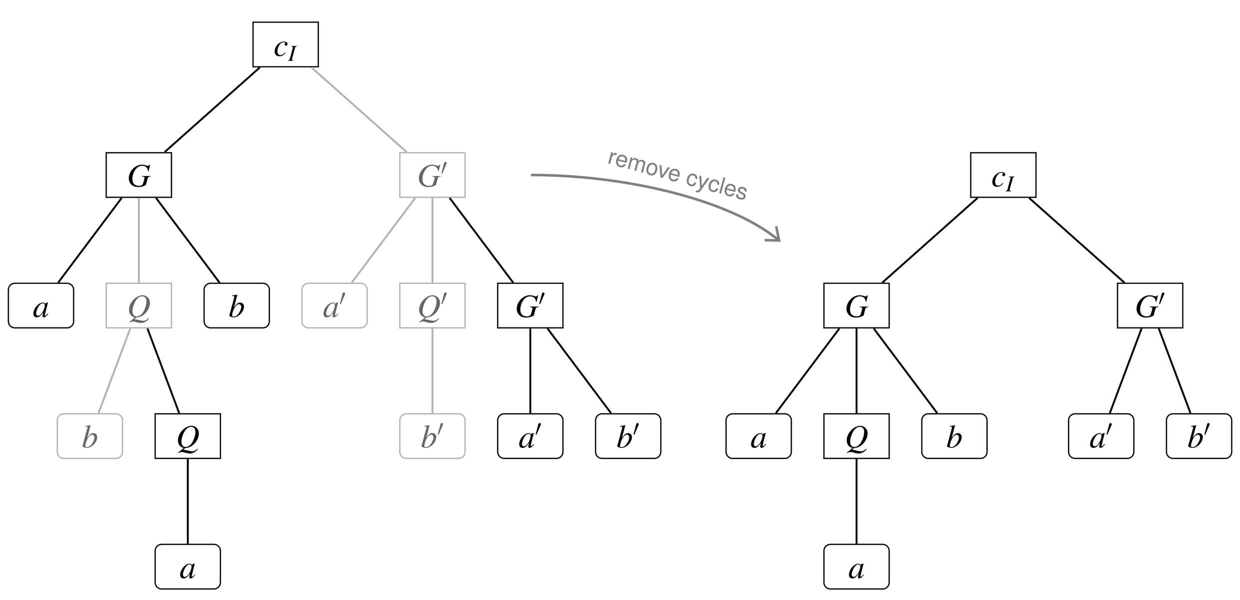
## Theorem

The plan existence problem is *undecidable* for HTN planning.

**Proof idea** (by Erol et al. [4]):

- the following question is undecidable: Given two CFGs, do their languages produce a common word?
- observe that the production rules of CFGs can be simulated by decomposition methods
- given two CFGs, construct a planning problem which has an HTN solution iff the languages of the two grammars have a non-empty intersection

**Figure 4:** *removing cycles from a decomposition tree*



## Decidability Criteria

Imposing certain restrictions on planning problems can make HTN planning decidable:

**Criterion 1:** Decomposition tree is acyclic
*Intuition:* search space becomes finite.

**Criterion 2:** All methods are totally ordered (cf. SHOP system [5])
*Intuition:* solution corresponds to an intersection of a regular grammar with one that is context-free (decidable problem).

**Criterion 3:** Methods contain at most one compound task (regular)
*Intuition:* the combinations of possible states before and after the abstract task are finite.

**Criterion 4:** Allow task insertion
*Intuition:* insertion makes cyclic method applications superfluous $\rightarrow$ minimal solution lengths are bounded like in classical planning.

## Theorem

The plan existence problem is *decidable* for hybrid planning.

**Proof Idea:**

- establish an upper bound on the size of shortest solutions
- enumerate all short task networks and check whether they are a solution
- if no solution has been found, then we know that no solution exists at all

## Proof — Bounding Decomposition

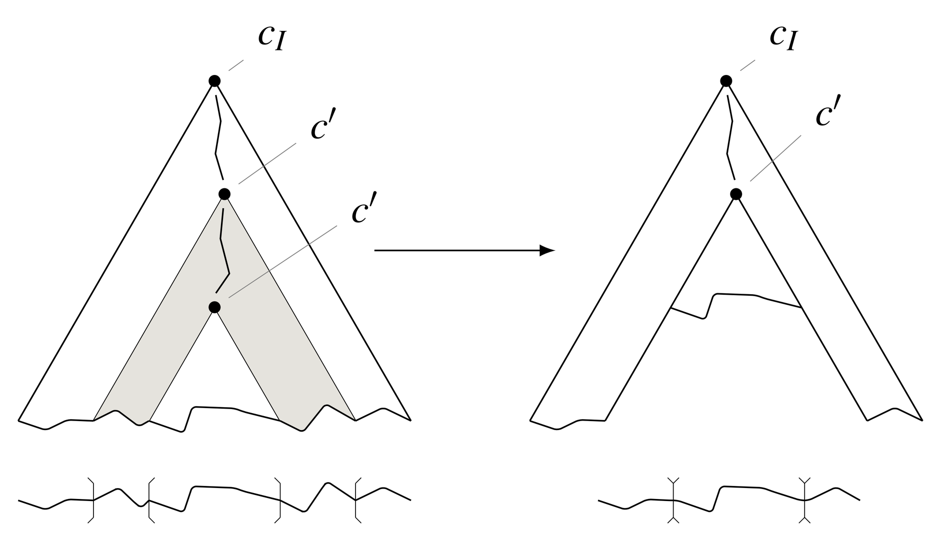
Any hybrid solution to $P$ can be constructed using at most $b^c$ decompositions where $b$ is the number of tasks inside the largest method. (maximum branching factor of the decomposition tree)

We apply the idea of the pumping lemma for context-free grammars to task decomposition:

1. remove all cycles from the decomposition tree
2. replace the removed elements using task and ordering insertion

Right after decomposition the intermediate task network contains at most $b^{|C|}$ tasks because the depth of the generating tree is limited by the number of compound tasks $|C|$.

**Figure 5:** *pumping down decompositions*



## Proof — Bounding Task Insertion

Given a task network $\mathrm{tn}$ with $n$ tasks, we have to insert at most $n2^{|V|}$ additional tasks to turn it into a solution, if this is possible at all.

- to obtain the bound, fix a (totally ordered) solution that contains $\mathrm{tn}$; thus it can be obtained from $\mathrm{tn}$ via insertion
- remove all task sequences that produce loops in the state space and that do not contain tasks from $\mathrm{tn}$
- we obtain a solution which contains at most $n(2^{|V|}+1)$ tasks

## Proof — Bounding Solutions

For every planning problem $P$, there exists a hybrid solution with at most $b^{|C|}(2^{|V|}+1)$ tasks if such a solution exists at all.

The bound follows from the bounds on task decomposition and the bound on task insertion.

## Conclusion & Discussion

- allowing task insertion makes HTN planning decidable resulting in the following complexity classes [1, 3, 4]:

| | Computational Complexity | | |
|---|---|---|---|
| | classical | hybrid | HTN |
| lifted | $\mathbf{EXPSPACE}$-complete | $\mathbf{EXPSPACE}$-hard | $\mathbf{RE}$ |
| propositional | $\mathbf{PSPACE}$-complete | $\mathbf{PSPACE}$-hard $\cap$ $\mathbf{EXPSPACE}$ | $\mathbf{RE}$ |

- one can think of HTN planning with insertion having a different meaning than HTN planning
  - HTN planning: specifies, what **must not** be in a plan
  - Hybrid planning: specifies, what **has to be** in a plan

## References


[1] Tom Bylander. Complexity results for planning. Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI 1991), pages 274–279, 1991.

[2] Kutluhan Erol, James Hendler, and Dana S. Nau. UMCP: A sound and complete procedure for hierarchical task-network planning. Proceedings of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS 1994), pages 249–254, 1994.

[3] Kutluhan Erol, Dana S. Nau, and V. S. Subrahmanian. Complexity, decidability and undecidability results for domain-independent planning. Artificial Intelligence, 76:75–88, 1995.

[4] Kutluhan Erol, James Hendler, and Dana S. Nau. Complexity results for HTN planning. Annals of Mathematics and Artificial Intelligence, 18(1):69–93, 1996.

[5] Dana S. Nau, Yue Cao, Amnon Lotem, and Héctor Muñoz-Avila. The SHOP Planning System. AI Magazine, 22(3):91–94, 2001.


ulm university universität uulm

Deutsche Forschungsgemeinschaft DFG