

# A Heuristic for Hybrid Planning With Preferences

Pascal Bercher and Susanne Biundo

Institute of Artificial Intelligence, Ulm University, Germany

FLAIRS 2012, Twenty-Fifth International Florida Artificial Intelligence Research Society Conference

## Abstract

We present a heuristic for hybrid planning with preferences on final states.

- It can be used for *hybrid planning* and for *POCL* planning,
- it reduces the problem of estimating the quality of a *task network* to the problem of estimating the quality of a *state*, and
- it performs a reachability analysis based on a planning graph [2].

## Hybrid Planning with Preferences (Problem Definition)

Hybrid planning [1] fuses HTN planning [3] with POCL planning [4].

A hybrid planning problem is a tuple  $(\mathcal{P}, \mathcal{T}_p, \mathcal{T}_c, \mathcal{M}, s_{init}, TN_{init}, g)$  with:

- $\mathcal{P}$  is a set of atomic, ground propositions,
- $\mathcal{T}_p, \mathcal{T}_c \subseteq 2^{\mathcal{P}} \times 2^{\mathcal{P}} \times 2^{\mathcal{P}}$  are sets of primitive and compound task schemata, resp.,
- $\mathcal{M} \subseteq \mathcal{T}_c \times \mathcal{T}_p$  is a set of decomposition methods,
- $s_{init} \in 2^{\mathcal{P}}$  is the initial state,
- $TN_{init} \in \mathcal{T}_p$  is the the initial partial plan, and
- $g \subseteq \mathcal{P}$  is the goal description.

The preferences on final states are given by weighted propositions:  
The function  $w : Pref \rightarrow \mathbb{R}$  maps preferences to their weight (or value).

**Definition (Task Network)** A task network  $TN$  is a tuple  $(T, \prec, CL)$  with:

- $T$ , a set of labeled tasks  $l:t$ ,  $l$  being a label symbol and  $t \in \mathcal{T}_p \cup \mathcal{T}_c$ ,
- $\prec$ , a partial order on  $T$ , and
- $CL$ , a set of causal links  $l' \rightarrow_{\phi} l$ .

The set of all task networks is referred to by  $\mathcal{TN}$ .

## Hybrid Planning with Preferences (Solution Criteria, -Quality)

A task network  $TN$  is a solution to a hybrid planning problem if and only if:

- $TN$  is a refinement of  $TN_{init}$  w.r.t. decomposition and insertions,
- $TN$  contains no compound tasks, and
- $TN$  has no open preconditions and no causal threats.

The quality of a solution is  $q(TN) := \sum_{p \in Pref \text{ with } TN \models p} w(p)$ . A solution  $TN_1$  is preferred over a solution  $TN_2$  if and only if  $q(TN_1) \geq q(TN_2)$ .

## Heuristic (Overview)

The heuristic consists of two steps:

1. domain transformation: transform a hybrid planning problem with a current *task network* into a relaxed classical planning problem with a current *state*
2. reachability analysis based on transformed problem

## References

- [1] Susanne Biundo and Bernd Schattenberg. From abstract crisis to concrete relief (a preliminary report on combining state abstraction and HTN planning). In *Proc. of the 6th European Conference on Planning (ECP 2001)*, pages 157–168, 2001.
- [2] Avrim L. Blum and Merrick L. Furst. Fast planning through planning graph analysis. *Artificial Intelligence*, 90:281–300, 1997.
- [3] Kutluhan Erol, James A. Hendler, and Dana S. Nau. UMCP: A sound and complete procedure for hierarchical task-network planning. In *Proc. of the 2nd International Conference on Artificial Intelligence Planning Systems (AIPS 1994)*, pages 249–254. AAAI Press, 1994.
- [4] David McAllester and David Rosenblitt. Systematic nonlinear planning. In *Proc. of the 9th National Conference on Artificial Intelligence (AAAI 1991)*, pages 634–639. AAAI Press, 1991.

## Domain Transformation

Given  $\pi = (\mathcal{P}, \mathcal{T}_p, \mathcal{T}_c, \mathcal{M}, s_{init}, TN_{init}, g)$  and a task network  $TN = (T, \prec, CL)$ , we construct a (relaxed) classical planning problem  $\pi'$ , s.t.:

- if  $TN$  can be refined to a solution of  $\pi$ , then  $\pi'$  is solvable as well, and
- every solution of  $\pi'$  is a refinement of  $TN$  w.r.t. its primitive tasks.

$\pi' = (\mathcal{P}', \mathcal{T}'_p, s'_{init}, g')$  is given by:

$$\bullet \mathcal{P}' := \mathcal{P} \cup \bigcup_{l:t \in T, t \in \mathcal{T}_p} \{l, \text{not-}l\}$$

$$\bullet \mathcal{T}'_p := \text{delete-relax}(\mathcal{T}_p) \cup \text{encode}(TN) \text{ with}$$

$$\text{delete-relax}(\mathcal{T}_p) := \{(\text{prec}, \text{add}, \emptyset) \mid (\text{prec}, \text{add}, \text{del}) \in \mathcal{T}_p\} \text{ and}$$

$$\text{encode}(TN) := \{\text{encode}(l:t) \mid l:t \in T \text{ and } t \in \mathcal{T}_p\}, \text{ where}$$

$$\text{encode}(l:(\text{pre}, \text{add}, \text{del})) :=$$

$$(\text{pre} \cup \{\text{not-}l\} \cup \{l' \mid l' \prec l \text{ or } l' \rightarrow_{\phi} l \in CL, l':t' \in T, \text{ and } t' \in \mathcal{T}_p\},$$

$$\text{add} \cup \{l\},$$

$$\text{del} \cup \{\text{not-}l\})$$

$$\bullet s'_{init} := s_{init} \cup \{\text{not-}l \mid \exists t \in \mathcal{T}_p, \text{ s.t. } l:t \in T\}$$

$$\bullet g' := g \cup \{l \mid \exists t \in \mathcal{T}_p, \text{ s.t. } l:t \in T\}$$

## Example

Let  $\pi = (\mathcal{P}, \mathcal{T}_p, \mathcal{T}_c, \mathcal{M}, s_{init}, TN_{init}, g)$  with  $\text{deliver}(loc_2) \in \mathcal{T}_c$  and  $d_1 := \text{drive}(truck_3, loc_1, loc_2) \in \mathcal{T}_p$ ,  $d_2 := \text{drive}(truck_3, loc_2, loc_3) \in \mathcal{T}_p$ , and  $TN$  given by:



Then,  $\pi' = (\mathcal{P}', \mathcal{T}'_p, s'_{init}, g')$  with:

- $\mathcal{P}' := \mathcal{P} \cup \{l_1, \text{not-}l_1, l_3, \text{not-}l_3\}$
- $\mathcal{T}'_p := \text{delete-relax}(\mathcal{T}_p) \cup$   
 $\{(\text{pre}(d_1) \cup \{\text{not-}l_1\}, \text{add}(d_1) \cup \{l_1\}, \text{del}(d_1) \cup \{\text{not-}l_1\})\} \cup$   
 $\{(\text{pre}(d_2) \cup \{\text{not-}l_3, l_1\}, \text{add}(d_2) \cup \{l_3\}, \text{del}(d_2) \cup \{\text{not-}l_3\})\}$
- $s'_{init} := s_{init} \cup \{\text{not-}l_1, \text{not-}l_3\}$
- $g' := g \cup \{l_1, l_3\}$

## Heuristic Calculation (Overview)

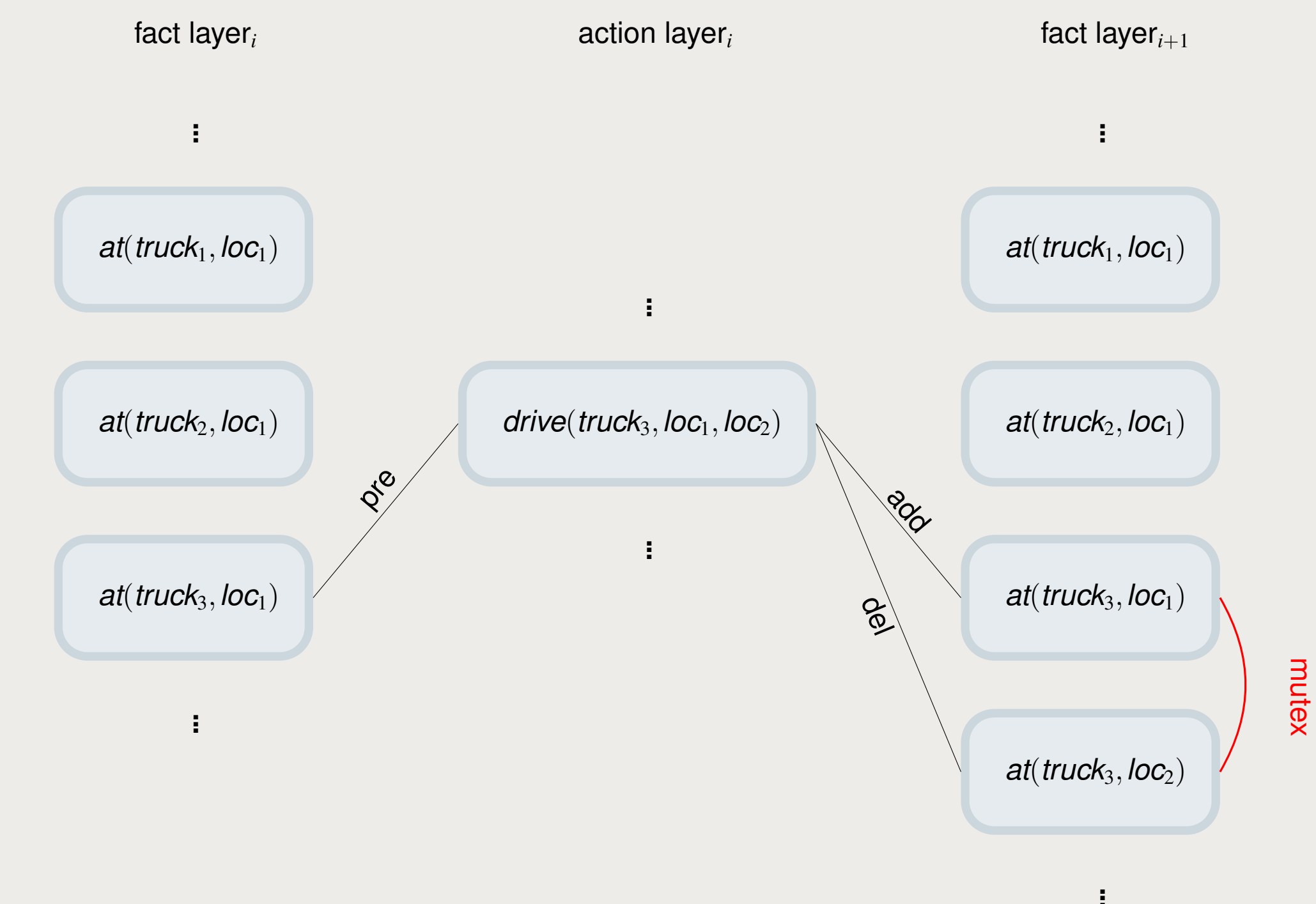
The heuristic value is calculated in two steps:

1. Calculate a partially relaxed planning graph until it has converged.
2. Use the mutex relations of the last fact layer to estimate the final plan quality based on the reachable (preferred) facts.

## Planning Graph Construction

The planning graph can be used for a (relaxed) reachability analysis.

## Example



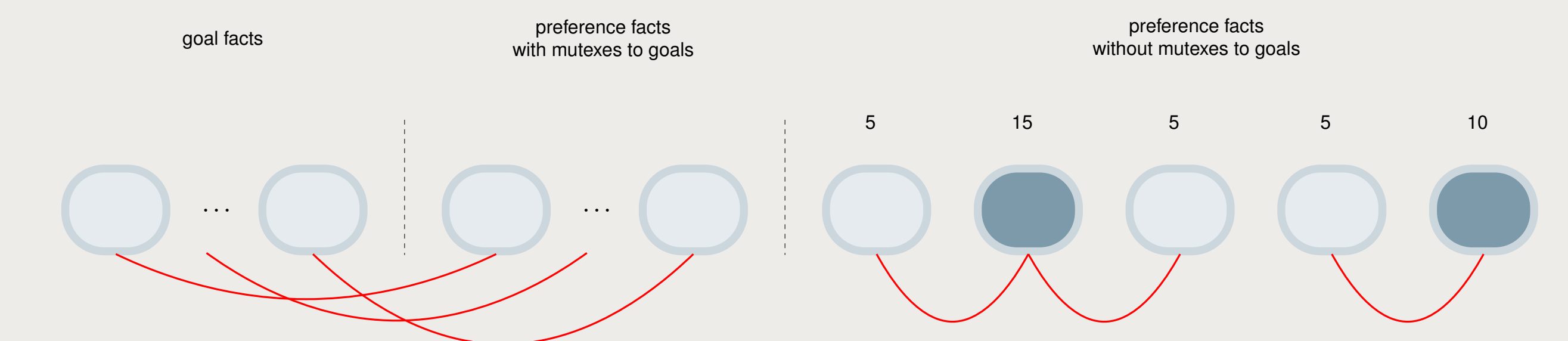
## Heuristic Calculation

The final fact layer (and its mutexes) are used to estimate the plan quality:

Let  $TN$  be the current task network,  $s'_{init}$  the new initial state of the transformed problem, and  $b$  a “mutex-respecting” truth assignment. Then,

$$h(TN) = h(s'_{init}) := \max_b \left( \sum_{\substack{p \in Pref, \\ b(p)=T}} w(p) \right)$$

## Example



Best possible quality of a task network featuring the given final fact layer is  $15 + 10 = 25$

