Domain Analysis: A Preprocessing Method that Reduces the Size of the Search Tree in Hybrid Planning

Michael Staud

StaudSoft UG (haftungsbeschränkt), Ulm, Germany

Objectives

Speeding up the planning process of an Hybrid Planner:

- We create new decomposition methods.
- They decompose an abstract task into a single (newly generated) primitive task.
- These new decomposition methods are created in a pre-processing step.

Introduction

We introduce a new method that can reduce the size of the search tree in hybrid planning. Hybrid planning fuses task insertion HTN planning with POCL planning. As planning is a computationally difficult problem, the size of the search tree can grow exponentially to the size of the problem.

We create so-called plan templates in a preprocessing step of the domain D. They can be used to replace an abstract task in a partial plan. This task then no longer needs to be decomposed.

Our contribution is an algorithm that generates replacement plans and placeholder tasks in a preprocessing step, which are then used during planning to reduce the size of the search tree. After a plan is found, the placeholder tasks are replaced by the replacement plans.

Planning Framework

A *hybrid planning domain* is a tuple $D = (T_a, T_p, M)$. T_a is the set of abstract tasks, T_p is the set of primitive tasks, and M is the set of all decomposition methods. Both primitive and abstract tasks are tuples $t(\tau) = \langle prec_t(\tau), \mathit{eff}_t(\tau)\rangle$ consisting of a precondition and an effect.

Partial plans are tuples $P = (PS, \prec, VC, CL)$ consisting of plan steps PS that are uniquely labeled tasks $l : t(\tau)$. The set $\prec$ contains the *ordering constraints* of the form $(l, l') \in PS \times PS$. The set VC contains the *unequal variable constraints* and the set CL contains the *causal links*. A causal link $l : t(\tau) \rightarrow_{\phi(\tau')} l' : t'(\tau')$ (shortened: $l \rightarrow_\phi l'$) links a precondition literal $\phi(\tau')$ of the plan step $l' : t'(\tau')$ to a unifiable effect of $l : t(\tau)$. Abstract tasks must be decomposed during the planning process using *decomposition methods*. A method $m = \langle t(\tau_m), P_m\rangle$ is a tuple that maps an abstract task $t(\tau_m)$ to a partial plan $P_m = (PS_m, \prec_m, VC_m, CL_m)$ that "implements" the task (Bercher et al., 2016, Def. 7).

Example

Abstract task `do_observation` from the domain `satellite2.pddl` :

```
(:task do_observation
 :parameters (?do_d - image_direction  ?do_m - mode)
 :effect (and (have_image ?do_d ?do_m))
)
```

Primitive task created from the replacement plan for `do_observation`:

```
(:action pt_do_observation_1
 :parameters (?var0 - image_direction ?var1 - mode ?var4 - satellite
     ?var5 - instrument ?var7 - calib_direction)
 :precondition (and (supports ?var5 ?var1) (on_board ?var5 ?var4)
     (calibration_target ?var5 ?var7) (pointing ?var4 ?var7)
     (power_avail ?var4))
 :effect (and (pointing ?var4 ?var0) (not (pointing ?var4 ?var7))
    (have_image ?var0 ?var1) (power_on ?var5) (not (power_avail ?var4))
    (calibrated ?var5))
)
```

Plan Template Generation

In the preprocessing step, a *plan template* is created for an abstract task $t_a(\tau_a) \in T_a$. This is done using a modified hybrid planner. The template is a tuple $P_T = \langle P_R(\tau), m_T(\tau), p_T(\tau)\rangle$ over the parameters τ ($\tau \supseteq \tau_a$). We need to generate:

- a *replacement plan* $P_R(\tau)$, that is created by using a modified hybrid planner. The initial partial plan for this planning problem only contains the initial conditions $t_0(\tau_0)$, the goal conditions $t_\infty(\tau_\infty)$ and the abstract task $t_a(\tau_a)$ for which the partial plan is created for. The planning process is also an optimization problem with an *objective function*.
- a primitive task $p_T(\tau)$, which is created out of the replacement plan. We add as preconditions all the effects of the initial state. And as effects we add all effects off all tasks in the replacement plan unless they are canceled out.
- and a method $m_T(\tau) = \langle t_a(\tau_a), P'\rangle$ that allows the planner to select the primitive task. It holds $P' = (\{l' : p_T(\tau)\}, \emptyset, \emptyset, \emptyset)$.

The new primitive task and the method are added to the domain. The new domain is called D_S.

Objective Function

The objective function evaluates a partial plan P. We define the following auxiliary functions:

- $ch(t_a(\tau), a)$: Returns 1 if the literal's predicate occurs as an effect in any decomposition of $t_a(\tau)$. Otherwise, it returns 0.
- $st_{ct}(A)$: It returns ∞ if a violation of the state constraints according to Gerevini and Schubert Gerevini and Schubert (1998) is found. Otherwise, it returns 0.
- $P(a \in S(t_a))$: Probability that the predicate of a is available as a precondition for t_a in a randomly selected problem in the domain D.

We tested two different objective functions $f_{t_a}(P_{opt}, t_a)$:

$f_1(P, t_a) = |prec_{t_0}(\tau)| + |PS|$

$$f_2^x(P, t_a) = \frac{|PS| + \sum\limits_{a \in prec_{t_0}(\tau)} + ch(t_a(\tau), a)x +}{st_{ct}(prec_{t_0}(\tau))}$$

Regarding f_2^x we used the two variants f_2^1 and $f_2^{P(a \in S(t_a))}$. We then select the plan template that maximizes the reduction of the search space. The others are *discarded*.

Selection of Abstract Tasks

To determine which abstract task is suitable for generating plan templates, we use the task decomposition graph Bercher, Keen, and Biundo (2014). For each task, we compute the set of mandatory M_{t_a} and optional tasks O_{t_a}. The mandatory tasks occur in every decomposition of a given abstract task t_a. The optional tasks occur in at least one decomposition.

In our experiments, the higher the ratio $|M_{t_a}|/|O_{t_a}|$, the more successful the generation of a plan template. Thus, we select the abstract task with the highest ratio.

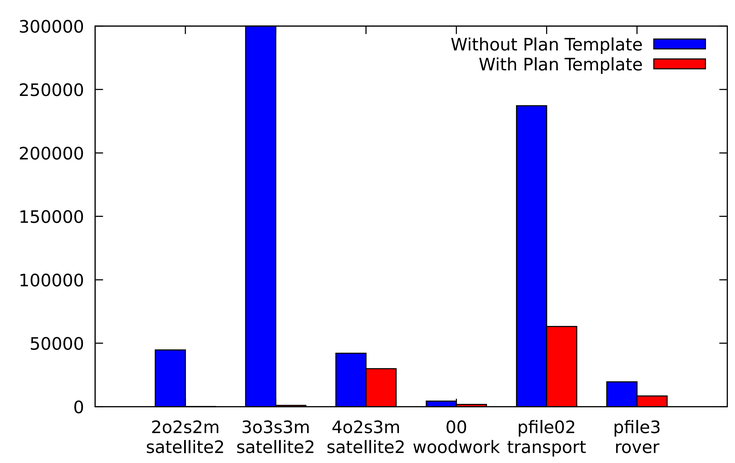
Replacement Process

The replacement process is a post-processing step performed after the solution $P_{Ds} = (PS_{Ds}, \prec_{Ds}, VC_{Ds}, CL_{Ds})$ has been generated in D_s. The goal is then to generate a solution $P_{Ds'} = (PS_{Ds'}, \prec_{Ds'}, VC_{Ds'}, CL_{Ds'})$ that is valid in the original domain D. For each plan step $l : p_T(\tau_t)$, which uses a primitive task from a plan template $P_T(\tau) = \langle P_R(\tau), m_T(\tau), p_T(\tau)\rangle$, $p_T(\tau) = \langle prec_{p_T}(\tau), \mathit{eff}_{p_T}(\tau)\rangle$ we perform the following steps:

- **Decomposition**: The plan step $l : p_T(\tau)$ is treated as an abstract task and decomposed using the method $m' = \langle p_T(\tau), P_R(\tau)\rangle$. Let $P_R = (PS_R, \prec_R, VC_R, CL_R)$.
- **Relinking**: After the decomposition, the plan steps $l_0, l_\infty \in PS_R$ are removed and their causal links are relinked. Let $l_0 \rightarrow_\phi l'$ be a casual link from l_0. And let $l'' \rightarrow_\phi l$ be a causal link to $l : p_T$. Then these two links are replaced by $l'' \rightarrow_\phi l'$. Similarly, a causal link of the form $l' \rightarrow_\phi l_\infty$ is treated.
- **Removing Causal Threats**: We remove causal threats introduced by the decomposition step. This is done by adding ordering constraints if possible. If this cannot be done we relink the causal links.

The plan will satisfy the solution criteria after the replacement process.

Results



Conclusion

We presented a new domain analysis method that can reduce the size of the search tree. We showed that a drastic reduction in the size of the search tree is empirically possible. This is even more effective when example problems are available to guide the search when generating the plan templates.

References

Bercher, P.; Höller, D.; Behnke, G.; and Biundo, S. 2016. More than a Name? On Implications of Preconditions and Effects of Compound HTN Planning Tasks. In *ECAI 2016*, 225–233. IOS Press.

Bercher, P.; Keen, S.; and Biundo, S. 2014. Hybrid Planning Heuristics Based on Task Decomposition Graphs. In *SoCS 2014*, 35–43. AAAI Press.

Gerevini, A.; and Schubert, L. 1998. Inferring State Constraints for Domain-Independent Planning. AAAI 98/IAAI 98, 905–912. AAAI Press.

University Ulm. 2019. PANDA Planning Domains and Problems. Https://www.uni-ulm.de/en/in/ki/-research/software/panda/, Accessed: 2020-10-01.