

Planning and Optimization

D3. Delete Relaxation: Finding Relaxed Plans

Malte Helmert and Gabriele Röger

Universität Basel

October 22, 2025

Planning and Optimization

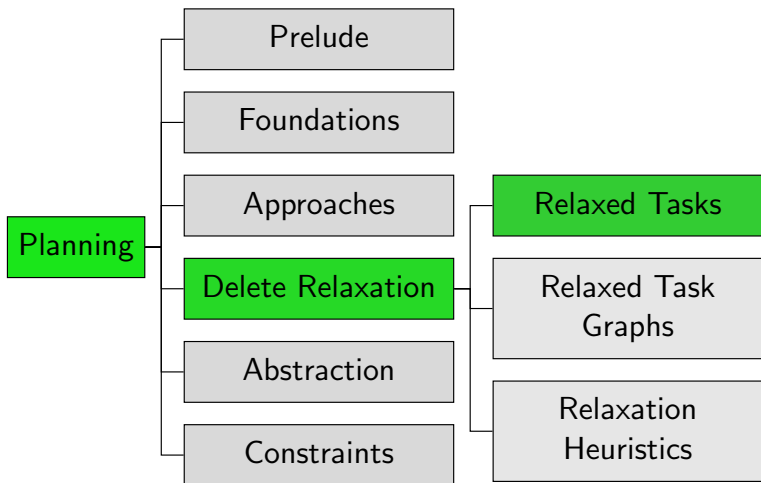
October 22, 2025 — D3. Delete Relaxation: Finding Relaxed Plans

D3.1 Greedy Algorithm

D3.2 Optimal Relaxed Plans

D3.3 Summary

Content of the Course



D3.1 Greedy Algorithm

The Story So Far

- ▶ A general way to come up with heuristics is to solve a **simplified** version of the real problem.
- ▶ **delete relaxation**: given a task in positive normal form, discard all delete effects
 - ▶ **relaxation lemma**: solutions for a state s also work for any dominating state s'
 - ▶ **monotonicity lemma**: $s[o]$ dominates s

Greedy Algorithm for Relaxed Planning Tasks

The relaxation and monotonicity lemmas suggest the following algorithm for solving relaxed planning tasks:

Greedy Planning Algorithm for $\langle V, I, O^+, \gamma \rangle$

$s := I$

$\pi^+ := \langle \rangle$

loop forever:

if $s \models \gamma$:

return π^+

else if there is an operator $o^+ \in O^+$ applicable in s
 with $s[o^+] \neq s$:

 Append such an operator o^+ to π^+ .

$s := s[o^+]$

else:

return unsolvable

Correctness of the Greedy Algorithm

The algorithm is **sound**:

- ▶ If it returns a plan, this is indeed a correct solution.
- ▶ If it returns “unsolvable”, the task is indeed unsolvable
 - ▶ Upon termination, there clearly is no relaxed plan from s .
 - ▶ By iterated application of the monotonicity lemma, s dominates I .
 - ▶ By the relaxation lemma, there is no solution from I .

What about **completeness** (termination) and **runtime**?

- ▶ Each iteration of the loop adds at least one atom to $on(s)$.
- ▶ This guarantees termination after at most $|V|$ iterations.
- ▶ Thus, the algorithm can clearly be implemented to run in polynomial time.
 - ▶ A good implementation runs in $O(\|\Pi\|)$.

Using the Greedy Algorithm as a Heuristic

We can apply the greedy algorithm within heuristic search for a general (non-relaxed) planning task:

- ▶ When evaluating a state s in progression search, solve relaxation of planning task with initial state s .
- ▶ When evaluating a subgoal φ in regression search, solve relaxation of planning task with goal φ .
- ▶ Set $h(s)$ to the cost of the generated relaxed plan.
 - ▶ in general not **well-defined**:
different choices of o^+ in the algorithm lead to different $h(s)$

Is this admissible/safe/goal-aware/consistent?

Properties of the Greedy Algorithm as a Heuristic

Is this an **admissible** heuristic?

- ▶ Yes if the relaxed plans are **optimal** (due to the plan preservation corollary).
- ▶ However, usually they are not, because the greedy algorithm can make poor choices of which operators to apply.

How hard is it to find **optimal** relaxed plans?

D3.2 Optimal Relaxed Plans

Optimal Relaxation Heuristic

Definition (h^+ heuristic)

Let $\Pi = \langle V, I, O, \gamma \rangle$ be a planning task in positive normal form with states S .

The **optimal delete relaxation heuristic** h^+ for Π

is the function $h : S \rightarrow \mathbb{R}_0^+ \cup \{\infty\}$

where $h(s)$ is the cost of an **optimal relaxed plan** for s ,
i.e., of an optimal plan for $\Pi_s^+ = \langle V, s, O^+, \gamma \rangle$.

(can analogously define a heuristic for regression)

admissible/safe/goal-aware/consistent?

The Set Cover Problem

Can we compute h^+ efficiently?

This question is related to the following problem:

Problem (Set Cover)

Given: a finite set U , a collection of subsets $C = \{C_1, \dots, C_n\}$ with $C_i \subseteq U$ for all $i \in \{1, \dots, n\}$, and a natural number K .

Question: Is there a set cover of size at most K , i.e., a subcollection $S = \{S_1, \dots, S_m\} \subseteq C$ with $S_1 \cup \dots \cup S_m = U$ and $m \leq K$?

The following is a classical result from complexity theory:

Theorem (Karp 1972)

The set cover problem is NP-complete.

Complexity of Optimal Relaxed Planning (1)

Theorem (Complexity of Optimal Relaxed Planning)

The BCPLANEX problem restricted to delete-relaxed planning tasks is NP-complete.

Proof.

For **membership in NP**, guess a plan and verify.

It is sufficient to check plans of length at most $|V|$ where V is the set of state variables, so this can be done in nondeterministic polynomial time.

For **hardness**, we reduce from the set cover problem. ...

Complexity of Optimal Relaxed Planning (2)

Proof (continued).

Given a set cover instance $\langle U, C, K \rangle$, we generate the following relaxed planning task $\Pi^+ = \langle V, I, O^+, \gamma \rangle$:

- ▶ $V = U$
- ▶ $I = \{v \mapsto \mathbf{F} \mid v \in V\}$
- ▶ $O^+ = \{\langle \top, \bigwedge_{v \in C_i} v, 1 \rangle \mid C_i \in C\}$
- ▶ $\gamma = \bigwedge_{v \in U} v$

If S is a set cover, the corresponding operators form a plan. Conversely, each plan induces a set cover by taking the subsets corresponding to the operators. There exists a plan of cost at most K iff there exists a set cover of size K .

Moreover, Π^+ can be generated from the set cover instance in polynomial time, so this is a polynomial reduction. □

D3.3 Summary

Summary

- ▶ Because of their monotonicity property, delete-relaxed tasks can be solved in **polynomial time** by a **greedy algorithm**.
- ▶ However, the solution quality of this algorithm is poor.
- ▶ For an informative heuristic, we would ideally want to find **optimal relaxed plans**.
- ▶ The solution cost of an optimal relaxed plan is the estimate of the h^+ heuristic.
- ▶ However, the bounded-cost plan existence problem for relaxed planning tasks is **NP-complete**.