

Planning and Optimization

D1. Delete Relaxation: Relaxed Planning Tasks

Malte Helmert and Gabriele Röger

Universität Basel

October 19, 2026

Planning and Optimization

October 19, 2026 — D1. Delete Relaxation: Relaxed Planning Tasks

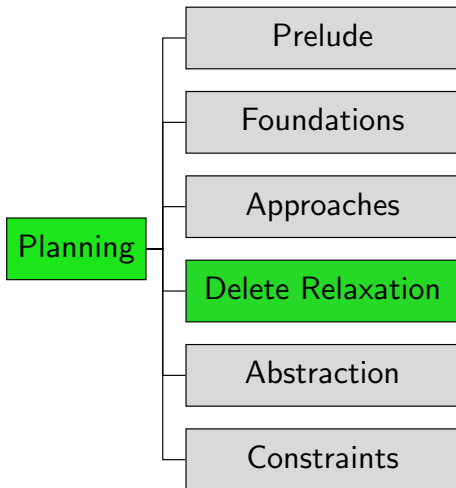
D1.1 Heuristics

D1.2 Coming Up with Heuristics

D1.3 Relaxed Planning Tasks

D1.4 Summary

Content of the Course



D1.1 Heuristics

Planning as Heuristic Search

- ▶ **Heuristic search** is the most common approach to planning.
- ▶ ingredients: **general search algorithm** + **heuristic**
- ▶ heuristic estimates cost from a given state to a given goal
 - ▶ **progression**: from varying states s to fixed goal γ
 - ▶ **regression**: from fixed initial state I to varying subgoals φ
- ▶ Over the next weeks, we study the main ideas behind heuristics for planning tasks.

Reminder: Heuristics

Need to Catch Up?

- ▶ We assume familiarity with heuristics and their properties:
 - ▶ **heuristic** $h : S \rightarrow \mathbb{R}_0^+ \cup \{\infty\}$
 - ▶ **perfect heuristic** h^* : $h^*(s)$ cost of optimal solution from s (∞ if unsolvable)
 - ▶ properties of heuristics h :
 - ▶ **safe**: $(h(s) = \infty \Rightarrow h^*(s) = \infty)$ for all states s
 - ▶ **goal-aware**: $h(s) = 0$ for all goal states s
 - ▶ **admissible**: $h(s) \leq h^*(s)$ for all states s
 - ▶ **consistent**: $h(s) \leq \text{cost}(o) + h(s')$ for all transitions $s \xrightarrow{o} s'$
 - ▶ connections between these properties
- ▶ If you are not familiar with these, we recommend Ch. B9–B10 of the [Foundations of Artificial Intelligence](https://dmi.unibas.ch/en/studies/computer-science/course-offer-spring-2026/13548-lecture-foundations-of-artificial-intelligence/) course:
<https://dmi.unibas.ch/en/studies/computer-science/course-offer-spring-2026/13548-lecture-foundations-of-artificial-intelligence/>

D1.2 Coming Up with Heuristics

A Simple Heuristic for Propositional Planning Tasks

STRIPS (Fikes & Nilsson, 1971) used the number of state variables that differ in current state s and a STRIPS goal $v_1 \wedge \dots \wedge v_n$:

$$h(s) := |\{i \in \{1, \dots, n\} \mid s \not\models v_i\}|.$$

Intuition: more satisfied goal atoms $\rightsquigarrow$ closer to the goal

$\rightsquigarrow$ **STRIPS heuristic (a.k.a. goal-count heuristic)**

Criticism of the STRIPS Heuristic

What is wrong with the STRIPS heuristic?

- ▶ quite **uninformative**:
the range of heuristic values in a given task is small;
typically, most successors have the same estimate
- ▶ very sensitive to **reformulation**:
can easily transform any planning task into an equivalent one
where $h(s) = 1$ for all non-goal states (how?)
- ▶ ignores almost all **problem structure**:
heuristic value does not depend on the set of operators!

↪ need a better, principled way of coming up with heuristics

Coming Up with Heuristics in a Principled Way

General Procedure for Obtaining a Heuristic

- ▶ **Simplify the problem**, for example by removing problem constraints.
- ▶ Solve the simplified problem (ideally optimally).
- ▶ Use the solution cost for the simplified problem as a heuristic for the real problem.

As heuristic values are computed for every generated search state, it is important that they can be computed **efficiently**.

Relaxing a Problem: Example

Example (Route Planning in a Road Network)

The road network is formalized as a weighted graph over points in the Euclidean plane. The weight of an edge is the **road distance** between two locations.

Example (Relaxation for Route Planning)

Use the **Euclidean distance** $\sqrt{|x_1 - x_2|^2 + |y_1 - y_2|^2}$ as a heuristic for the road distance between $\langle x_1, y_1 \rangle$ and $\langle x_2, y_2 \rangle$. This is a **lower bound** on the road distance ($\rightsquigarrow$ admissible).

$\rightsquigarrow$ We drop the constraint of having to travel on roads.

Planning Heuristics: Main Concepts

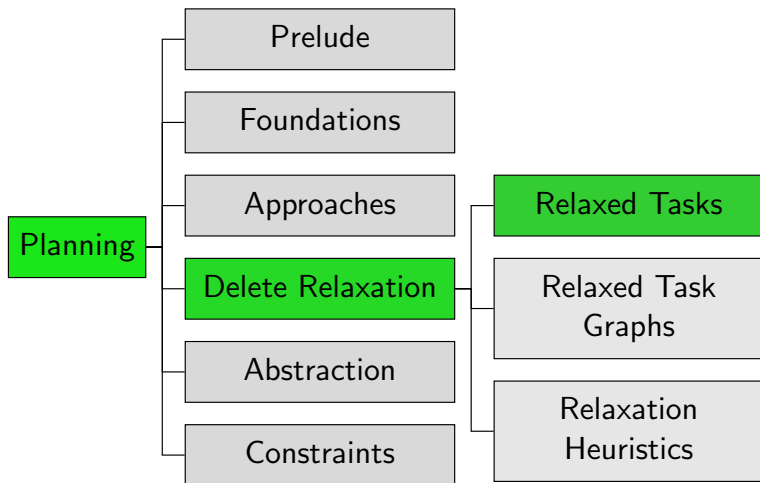
Major ideas for heuristics in the planning literature:

- ▶ delete relaxation $\rightsquigarrow$ Part D
- ▶ abstraction $\rightsquigarrow$ Part E
- ▶ critical paths $\rightsquigarrow$ not considered in this course
- ▶ landmarks $\rightsquigarrow$ Part F
- ▶ network flows $\rightsquigarrow$ Part F
- ▶ potential heuristics $\rightsquigarrow$ Part F

We will consider most of them in this course.

D1.3 Relaxed Planning Tasks

Content of the Course



Delete Relaxation: Idea

In **positive normal form** (Chapter B5, remember?), good and bad effects are easy to distinguish*:

- ▶ Effects that make state variables true are good (**add effects**).
- ▶ Effects that make state variables false are bad (**delete effects**).

Idea of **delete relaxation heuristics**: ignore all delete effects.

(*) with a small caveat regarding conditional effects

Delete-Relaxed Planning Tasks

Definition (Delete Relaxation of Operators)

The **delete relaxation** o^+ of an operator o in positive normal form is the operator obtained by replacing all negative effects $\neg a$ within $\text{eff}(o)$ by the do-nothing effect $\top$.

Definition (Delete Relaxation of Propositional Planning Tasks)

The **delete relaxation** Π^+ of a propositional planning task $\Pi = \langle V, I, O, \gamma \rangle$ in positive normal form is the planning task $\Pi^+ := \langle V, I, \{o^+ \mid o \in O\}, \gamma \rangle$.

Definition (Delete Relaxation of Operator Sequences)

The **delete relaxation** of an operator sequence $\pi = \langle o_1, \dots, o_n \rangle$ is the operator sequence $\pi^+ := \langle o_1^+, \dots, o_n^+ \rangle$.

Note: “delete” is often omitted: **relaxation**, **relaxed**

Relaxed Planning Tasks: Terminology

- ▶ Planning tasks in positive normal form without delete effects are called **relaxed planning tasks**.
- ▶ Plans for relaxed planning tasks are called **relaxed plans**.
- ▶ If Π is a planning task in positive normal form and π^+ is a plan for Π^+ , then π^+ is called a **relaxed plan for Π** .

D1.4 Summary

Summary

- ▶ A general way to come up with heuristics:
solve a **simplified** version of the real problem,
for example by removing problem constraints.
- ▶ **delete relaxation**: given a task in positive normal form,
discard all delete effects