

Foundations of Artificial Intelligence

38. Automated Planning: Landmarks

Malte Helmert

University of Basel

May 8, 2019

Foundations of Artificial Intelligence

May 8, 2019 — 38. Automated Planning: Landmarks

38.1 Delete Relaxation

38.2 Landmarks

38.3 Exploiting Landmarks

38.4 Summary

Planning Heuristics

We discuss **three basic ideas** for general heuristics:

- ▶ Delete Relaxation
- ▶ Abstraction
- ▶ **Landmarks** ↵ this and next chapter

Basic Idea: Landmarks

landmark = something (e.g., an action) that must be part of **every solution**

Estimate solution costs by the number of unachieved landmarks.

Automated Planning: Overview

Chapter overview: automated planning

- ▶ 33. Introduction
- ▶ 34. Planning Formalisms
- ▶ 35.–36. Planning Heuristics: Delete Relaxation
- ▶ 37. Planning Heuristics: Abstraction
- ▶ 38.–39. Planning Heuristics: Landmarks
 - ▶ **38. Landmarks**
 - ▶ 39. Landmark Heuristics

38.1 Delete Relaxation

Landmarks and Delete Relaxation

- ▶ In this chapter, we discuss a further technique to compute planning heuristics: **landmarks**.
- ▶ We restrict ourselves to **delete-free** planning tasks:
 - ▶ For a STRIPS task Π , we compute its delete relaxed task Π^+ ,
 - ▶ and then apply landmark heuristics on Π^+ .
- ▶ Hence the objective of our landmark heuristics is to approximate the **optimal delete relaxed heuristic** h^+ as accurately as possible.
- ▶ More advanced landmark techniques work directly on general planning tasks.

German: Landmarke

Delete-Free STRIPS planning tasks

reminder:

Definition (delete-free STRIPS planning task)

A **delete-free STRIPS planning task** is a 4-tuple $\Pi^+ = \langle V, I, G, A \rangle$ with the following components:

- ▶ V : finite set of **state variables**
- ▶ $I \subseteq V$: the **initial state**
- ▶ $G \subseteq V$: the set of **goals**
- ▶ A : finite set of **actions**, where for every $a \in A$, we define
 - ▶ $pre(a) \subseteq V$: its **preconditions**
 - ▶ $add(a) \subseteq V$: its **add effects**
 - ▶ $cost(a) \in \mathbb{N}_0$: its **cost**

denoted as $pre(a) \xrightarrow{cost(a)} add(a)$ (omitting set braces)

Delete-Free STRIPS Planning Task in Normal Form

A delete-free STRIPS planning task $\langle V, I, G, A \rangle$ is in **normal form** if

- ▶ I consists of exactly one element i : $I = \{i\}$
- ▶ G consists of exactly one element g : $G = \{g\}$
- ▶ Every action has at least one precondition.

German: Normalform

Every task can easily be transformed into an equivalent task in normal form. (How?)

- ▶ In the following, we assume tasks in normal form.
- ▶ Describing A suffices to describe overall task:
 - ▶ V are the variables mentioned in A 's actions.
 - ▶ always $I = \{i\}$ and $G = \{g\}$
- ▶ In the following, we only describe A .

Example: Delete-Free Planning Task in Normal Form

Example

actions:

- ▶ $a_1 = i \xrightarrow{3} x, y$
- ▶ $a_2 = i \xrightarrow{4} x, z$
- ▶ $a_3 = i \xrightarrow{5} y, z$
- ▶ $a_4 = x, y, z \xrightarrow{0} g$

optimal solution to reach $\{g\}$ from $\{i\}$:

- ▶ **plan:** a_1, a_2, a_4
- ▶ **cost:** $3 + 4 + 0 = 7$ ($= h^+(\{i\})$ because plan is **optimal**)

38.2 Landmarks

Landmarks

Definition (landmark)

A **landmark** of a planning task Π is a set of actions L such that **every plan** must contain an action from L .

The **cost** of a landmark L , $cost(L)$ is defined as $\min_{a \in L} cost(a)$.

↔ landmark cost corresponds to (very simple) admissible heuristic

- ▶ Speaking more strictly, landmarks as considered in this course are called **disjunctive action landmarks**.
- ▶ other kinds of landmarks exist (fact landmarks, formula landmarks, ...)

German: disjunctive Aktionslandmarke, Faktlandmarke, Formellandmarke

Example: Landmarks

Example

actions:

- ▶ $a_1 = i \xrightarrow{3} x, y$
- ▶ $a_2 = i \xrightarrow{4} x, z$
- ▶ $a_3 = i \xrightarrow{5} y, z$
- ▶ $a_4 = x, y, z \xrightarrow{0} g$

some landmarks:

- ▶ $A = \{a_4\}$ (cost 0)
- ▶ $B = \{a_1, a_2\}$ (cost 3)
- ▶ $C = \{a_1, a_3\}$ (cost 3)
- ▶ $D = \{a_2, a_3\}$ (cost 4)
- ▶ also: $\{a_1, a_2, a_3\}$ (cost 3), $\{a_1, a_2, a_4\}$ (cost 0), ...

Overview: Landmarks

in the following:

► **exploiting landmarks:**

How can we compute an accurate heuristic for a given set of landmarks?

~~ this chapter

► **finding landmarks:**

How can we find landmarks?

~~ next chapter

► **LM-cut heuristic:**

an algorithm to find landmarks and exploit them as heuristic

~~ next chapter

38.3 Exploiting Landmarks

Exploiting Landmarks

Assume the set of landmarks $\mathcal{L} = \{A, B, C, D\}$.

How to use \mathcal{L} for computing heuristics?

► **sum** the costs: $0 + 3 + 3 + 4 = 10$

~~ not admissible!

► **maximize** the costs: $\max \{0, 3, 3, 4\} = 4$

~~ usually yields a weak heuristic

► **better:** hitting sets or cost partitioning

German: Hitting-Set, Kostenpartitionierung

Hitting Sets

Definition (hitting set)

given: finite **support set** X , **family of subsets** $\mathcal{F} \subseteq 2^X$,
cost $c : X \rightarrow \mathbb{R}_0^+$

hitting set:

- subset $H \subseteq X$ that “hits” all subsets in \mathcal{F} :
 $H \cap S \neq \emptyset$ for all $S \in \mathcal{F}$

- **cost** of H : $\sum_{x \in H} c(x)$

minimum hitting set (MHS):

- hitting set with minimal cost
- “classical” NP-complete problem (Karp, 1972)

Example: Hitting Sets

Example

$$X = \{a_1, a_2, a_3, a_4\}$$

$$\mathcal{F} = \{A, B, C, D\}$$

with $A = \{a_4\}$, $B = \{a_1, a_2\}$, $C = \{a_1, a_3\}$, $D = \{a_2, a_3\}$

$$c(a_1) = 3, c(a_2) = 4, c(a_3) = 5, c(a_4) = 0$$

minimum hitting set: $\{a_1, a_2, a_4\}$ with cost $3 + 4 + 0 = 7$

Hitting Sets for Landmarks

idea: landmarks are interpreted as instance of **minimum hitting set**

Definition (hitting set heuristic)

Let \mathcal{L} be a set of landmarks for a delete-free planning task in normal form with actions A , action costs $cost$ and initial state I .

The **hitting set heuristic** $h^{MHS}(I)$ is defined as the minimal solution cost for the minimum hitting set instance with support set A , family of subsets \mathcal{L} and costs $cost$.

Proposition (Hitting Set Heuristic is Admissible)

The minimum hitting set heuristic h^{MHS} is admissible.

Why?

Approximation of h^{MHS}

- As computing minimal hitting sets is NP-hard, we want to approximate h^{MHS} in polynomial time.

Optimal Cost Partitioning (Karpas & Domshlak, 2009)

idea: Construct a **linear program** (LP) for \mathcal{L} .

- rows (constraints) correspond to actions
- columns (variables) correspond to landmarks
- entries: 1 if row action is contained in column landmark; 0 otherwise
- objective: maximize sum of variables

heuristic value h^{OCP} (optimal cost partitioning):

objective value of LP

Example: Optimal Cost Partitioning

Example

$$cost(a_1) = 3, cost(a_2) = 4, cost(a_3) = 5, cost(a_4) = 0$$

$$\mathcal{L} = \{A, B, C, D\}$$

with $A = \{a_4\}$, $B = \{a_1, a_2\}$, $C = \{a_1, a_3\}$, $D = \{a_2, a_3\}$

LP: maximize $a + b + c + d$ subject to $a, b, c, d \geq 0$ and

$$\begin{array}{rcl} b + c & \leq & 3 & a_1 \\ b + d & \leq & 4 & a_2 \\ c + d & \leq & 5 & a_3 \\ a & \leq & 0 & a_4 \\ \hline A & B & C & D \end{array}$$

solution: $a = 0, b = 1, c = 2, d = 3 \rightsquigarrow h^{OCP}(I) = 6$

Relationship of Heuristics

Proposition (h^{OCP} vs. h^{MHS})

Let \mathcal{L} be a set of landmarks for a planning task with initial state I .

Then $h^{\text{OCP}}(I) \leq h^{\text{MHS}}(I) \leq h^+(I)$

The heuristic h^{OCP} can be computed in polynomial time because linear programs can be solved in polynomial time.

38.4 Summary

Summary

- ▶ **Landmarks** are action sets such that every plan must contain at least one of the actions.
- ▶ **Hitting sets** yield the most accurate heuristic for a given set of landmarks, but the computation is NP-hard.
- ▶ **Optimal cost partitioning** is a polynomial approach for the computation of informative landmark heuristics.