

# Principles of AI Planning

## 12. The LM-cut heuristic

Albert-Ludwigs-Universität Freiburg



**UNI  
FREIBURG**

Bernhard Nebel and Robert Mattmüller

December 19th, 2014



## The LM-cut heuristic

- Motivation
- Definitions
- Finding and exploiting landmarks
- Admissibility
- Summary

# The LM-cut heuristic



- **RPG-based relaxation heuristics** seen so far,
  - either **admissible**, but **not very informative** ( $h_{\max}$ ),
  - or **quite informative**, but **not admissible** ( $h_{\text{add}}$ ,  $h_{\text{sa}}$ ,  $h_{\text{FF}}$ ).
- $\rightsquigarrow$  **no useful relaxation heuristic for optimal planning yet.**
- This chapter: **informative admissible relaxation heuristic** ( $h_{\text{LM-cut}}$ ).
- $h_{\text{LM-cut}}$  one of the most informative admissible domain-independent heuristics currently known.



## Combination of several ideas:

- **Delete relaxation**
  - Already known from Chapter 7.
  - No repeated discussion in current chapter necessary.
- **Landmarks**
  - The central concept behind  $h_{\text{LM-cut}}$ .
  - Discussed first in this chapter.
- **Cost partitioning**
  - Only relevant in the non-unit-cost setting.
  - Discussed towards the end of this chapter.

The LM-cut  
heuristic

Motivation

Definitions

Finding and  
exploiting  
landmarks

Admissibility

Summary



Let  $\Pi$  be an SAS<sup>+</sup> planning task and  $s$  a state from  $\Pi$ .

Assume we know the following:

- In each plan starting in  $s$ , at least one of the operators  $o_1$  and  $o_2$  is applied.
- In each plan starting in  $s$ , at least one of the operators  $o_3$  and  $o_4$  is applied.
- In each plan starting in  $s$ , the operator  $o_5$  is applied.
- In each plan starting in  $s$ , the operator  $o_6$  is applied.
- Operators  $o_1, o_2, o_3, o_4, o_5$ , and  $o_6$  are pairwise different.

**Question:** Does this give us a lower bound on  $h^*(s)$ ?

**Answer:** Yes! The number of **landmarks**, i.e.,  $h^*(s) \geq 4$ .

The LM-cut  
heuristic

Motivation

Definitions

Finding and  
exploiting  
landmarks

Admissibility

Summary



Let  $\Pi$  be an SAS<sup>+</sup> planning task and  $s$  a state from  $\Pi$ .

Assume we know the following:

- In each plan starting in  $s$ , at least one of the operators  $o_1$  and  $o_2$  is applied.
- In each plan starting in  $s$ , at least one of the operators  $o_3$  and  $o_4$  is applied.
- In each plan starting in  $s$ , the operator  $o_5$  is applied.
- In each plan starting in  $s$ , the operator  $o_6$  is applied.
- Operators  $o_1, o_2, o_3, o_4, o_5$ , and  $o_6$  are pairwise different.

**Question:** Does this give us a lower bound on  $h^*(s)$ ?

**Answer:** Yes! The number of **landmarks**, i.e.,  $h^*(s) \geq 4$ .

The LM-cut  
heuristic

Motivation

Definitions

Finding and  
exploiting  
landmarks

Admissibility

Summary



- Technique for derivation of heuristic: **landmarks**.
- Question: How to compute suitable landmarks?
- For now (as long as we only consider unit-cost actions) **suitable** landmarks means **disjoint** landmarks.

Counterexample for non-disjoint landmarks: Knowing that

- in each plan starting in  $s$ , at least one of the operators  $o_1$  and  $o_2$  is applied, and
- in each plan starting in  $s$ , at least one of the operators  $o_2$  and  $o_3$  is applied,

does **not** imply that  $h^*(s) \geq 2$ , since the one-step action sequence  $o_2$  might be a plan for  $s$ .

The LM-cut  
heuristic

Motivation

Definitions  
Finding and  
exploiting  
landmarks  
Admissibility  
Summary



## Definition (Landmark)

A **landmark** of an SAS<sup>+</sup> planning task  $\Pi$  is a set of actions  $L$  such that **each plan** for  $\Pi$  contains at least one action from  $L$ . A landmark  $L$  for  $\Pi$  is **minimal** if no  $L' \subsetneq L$  is a landmark for  $\Pi$ .

**Note:** Landmarks in this sense are also called **disjunctive action landmarks**.

## Theorem

*Let  $\Pi$  with initial state  $I$  be an SAS<sup>+</sup> planning task. If there are  $n$  **disjoint** landmarks for  $\Pi$ , then  $h(I) = n$  is an admissible heuristic estimate for state  $I$ .*

## Proof.

Obvious. □

The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

Admissibility

Summary



## Example

$\langle A, I, \{o_1, o_2, o_3, o_4, o_5\}, \gamma \rangle$  with

$$A = \{a, b, c, d, e, f, g\} \quad I = \{a \mapsto 1\} \cup \{x \mapsto 0 \mid x \neq a\}$$

$$o_1 = \langle a, b \wedge c \rangle$$

$$o_2 = \langle a, c \wedge d \rangle$$

$$o_3 = \langle a, d \wedge e \rangle$$

$$o_4 = \langle a, e \wedge b \rangle$$

$$o_5 = \langle a, f \rangle$$

$$o_6 = \langle b \wedge c \wedge d \wedge e \wedge f, g \rangle$$

$$\gamma = g$$

(Minimal) landmarks:

$\{o_1, o_2\}$  (because of  $c$ ),

$\{o_2, o_3\}$  (because of  $d$ ),

$\{o_3, o_4\}$  (because of  $e$ ),

$\{o_4, o_1\}$  (because of  $b$ ),

$\{o_5\}$  (because of  $f$ ),

$\{o_6\}$  (because of  $g$ )

The LM-cut  
heuristic

Motivation

**Definitions**

Finding and  
exploiting  
landmarks

Admissibility

Summary



## Example (ctd.)

But at most four disjoint landmarks, e.g.,  
 $\{o_1, o_2\}, \{o_3, o_4\}, \{o_5\}, \{o_6\}$ .

$\rightsquigarrow h_{LM}(I) = 4$  is admissible.



## Theorem

*Let  $\Pi$  be an SAS<sup>+</sup> planning task, and let  $\Pi^+$  be its delete relaxation. Let  $L^+ = \{o^+ \mid o \in L\}$  be a landmark for  $\Pi^+$ . Then  $L$  is also a landmark for  $\Pi$ .*

## Proof.

Let  $L^+$  be a landmark for  $\Pi^+$ . Then every plan  $\pi^+$  for  $\Pi^+$  uses some action  $o^+ \in L^+$ .

Let  $\pi'$  be some plan for  $\Pi$ . We need to show that  $\pi'$  uses some action  $o \in L$ . Since  $\pi'$  is a plan for  $\Pi$ , also  $\pi'^+$  is a plan for  $\Pi^+$ . By assumption,  $\pi'^+$  must use some action  $o^+ \in L^+$ . But then,  $\pi'$  uses action  $o \in L$ . □

The LM-cut heuristic

Motivation

**Definitions**

Finding and exploiting landmarks

Admissibility

Summary



## Theorem

*Let  $\Pi$  be an SAS<sup>+</sup> planning task, and let  $\Pi^+$  be its delete relaxation. Let  $L^+ = \{o^+ \mid o \in L\}$  be a landmark for  $\Pi^+$ . Then  $L$  is also a landmark for  $\Pi$ . □*

↪ It is sufficient to search for landmarks in the delete relaxation. This will only lead to too few discovered landmarks, not to too many.

↪ Admissibility of the heuristic will be preserved.

The LM-cut  
heuristic

Motivation

**Definitions**

Finding and  
exploiting  
landmarks

Admissibility

Summary



For the rest of this chapter, we assume **delete-free** planning tasks  $\Pi = \Pi^+$  and search for landmarks for  $\Pi^+$ , which gives us a good approximation of the **optimal delete relaxation heuristic**  $h^+$ .



### Naive approach:

- 1 Compute set  $\mathcal{L} = \{L_1, \dots, L_n\}$  of **all** minimal landmarks of planning task  $\Pi$ .
- 2 Compute a cardinality-maximal subset  $\mathcal{L}' \subseteq \mathcal{L}$  such that all  $L_i, L_j \in \mathcal{L}'$ ,  $L_i \neq L_j$ , are pairwise disjoint, and return their number,  $|\mathcal{L}'|$ .

**Drawbacks of naive approach:** Both steps too complicated.

### Simpler incomplete approach:

Compute set  $\mathcal{L} = \{L_1, \dots, L_n\}$  of **some disjoint** minimal landmarks for  $\Pi$  **incrementally**.

- Compute some landmark  $L_1$ .
- When computing  $L_{i+1}$ , only consider candidates that are disjoint from all previous landmarks  $L_1, \dots, L_i$ .
- Stop when no more such landmarks exist.

The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

Admissibility

Summary



We implement the simpler approach by exploiting a relationship between landmarks and cuts in certain graphs:

- **Assumption:** STRIPS tasks with action costs 0 or 1.
- When computing landmark  $L_{j+1}$ , an action  $o$  costs zero if:
  - it is a dummy action  $o_s$  constructing the initial state from the unique initial proposition  $s$ ,
  - it is a dummy action  $o_t$  constructing the unique dummy goal proposition  $t$  from the actual goal propositions, or
  - it **has already been included in one of the previous landmarks**  $L_1, \dots, L_j$ , i.e., it has already been accounted for in the heuristic computation.

The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

Admissibility

Summary



- To that end, in the algorithm we will present, action cost values will be iteratively decremented.
  - In the first iteration, we have action costs  $c_1(o_s) = c_1(o_t) = 0$ , and  $c_1(o) = 1$  for all other actions  $o$ .
  - Cost functions in later iterations  $i + 1$  are denoted by  $c_{i+1}$  and will differ from  $c_i$  in that costs of actions used in  $L_i$  are set to zero.

The LM-cut heuristic

Motivation

**Definitions**

Finding and exploiting landmarks

Admissibility

Summary





## Definition (Precondition-choice function)

A **precondition-choice function (pcf)** is a function  $D$  that maps each action into one of its preconditions.

(We assume that each action has at least one precondition.)

## Definition (Justification graph)

The **justification graph** for a pcf  $D$ , denoted by  $G(D)$ , is a directed graph whose vertices are the propositions and which has an edge  $(p, q)$  labeled with  $o$  iff the action  $o$  adds  $q$  and  $D(o) = p$ .

The LM-cut heuristic

Motivation

**Definitions**

Finding and exploiting landmarks

Admissibility

Summary



## Definition (Cut)

For two nodes  $\mathbf{s}$  and  $\mathbf{t}$  in a justification graph, an **s-t cut** in that justification graph is a subset  $C$  of its edges such that all paths from  $\mathbf{s}$  to  $\mathbf{t}$  use an edge from  $C$ .

When  $\mathbf{s}$  and  $\mathbf{t}$  are clear, we simply call  $C$  a cut.

## Theorem (Cuts correspond to landmarks)

*Let  $C$  be a cut in a justification graph for an arbitrary pcf.  
Then the edge labels for  $C$  are a landmark.* □

The LM-cut  
heuristic

Motivation

**Definitions**

Finding and  
exploiting  
landmarks

Admissibility

Summary



## Definition ( $h_{\max}$ costs of atoms)

Given a fixed initial state  $s$  and an action cost function  $c$ , the  $h_{\max}$  cost of an atom  $a$ , denoted by  $h_{\max}^c(a)$ , is the value the RPG proposition node for atom  $a$  in the last RPG layer is labeled with after the RPG computation (with layer 0 initialized with state  $s$  and action costs given by  $c$ ) has converged/stabilized.

Intuitively,  $h_{\max}^c(a)$  is the cost of making  $a$  true under parallel relaxed semantics, maximizing over precondition costs. For unit-costs tasks,  $h_{\max}^c(a)$  would be the index of the first RPG layer in the RPG seeded with  $s$  where  $a$  becomes true.

The LM-cut heuristic

Motivation

**Definitions**

Finding and exploiting landmarks

Admissibility

Summary



- In general **exponentially many pcfs**, i.e., we cannot compute all relevant landmarks.
  - The **LM-cut heuristic** is a method to compute pcfs and cuts in a **goal-directed** way.
  - Efficient partitioning of actions into cuts.
- ~> **currently best admissible planning heuristic**



Initialize  $h = 0$  and  $i = 1$ .

Step 1. Compute  $h_{\max}^{c_i}(a)$  values for every atom  $a \in A$ .  
Terminate if  $h_{\max}^{c_i}(\mathbf{t}) = 0$ .

Step 2. Compute pcf  $D_i$ : Modify actions by keeping only one proposition in the precondition of each action: a proposition maximizing  $h_{\max}^{c_i}$ , breaking ties arbitrarily.

Step 3. Construct justification graph  $G_i$  of  $D_i$ : Vertices are the propositions; for each action  $o = \langle p, q_1 \wedge \dots \wedge q_k \rangle$  and each  $j = 1, \dots, k$ , there is an edge from  $p$  to  $q_j$  with cost  $c_i(o)$  and label  $o$ .

Step 4. ...

The LM-cut  
heuristic

Motivation

Definitions

Finding and  
exploiting  
landmarks

Admissibility

Summary

# Pseudocode of LM-cut heuristic (ctd.)



- Step 4. **Construct an s-t-cut**  $C_i = (V_i^0, V_i^* \cup V_i^b)$  of  $G_i$  as follows:  $V_i^*$  contains all propositions from which  $\mathbf{t}$  can be reached through a zero-cost path,  $V_i^0$  contains all propositions reachable from  $\mathbf{s}$  without passing through some propositions in  $V_i^*$ , and  $V_i^b$  contains all remaining propositions. Clearly,  $\mathbf{s} \in V_i^0$  and  $\mathbf{t} \in V_i^*$ .
- Step 5. **Determine disjunctive action landmark:** Let  $L_i$  be the set of labels of the edges that cross the cut  $C_i$  (i.e., lead from  $V_i^0$  to  $V_i^*$ ).
- Step 6. **Decrease action costs:** Define  $c_{i+1}(o) := c_i(o)$  if  $o \notin L_i$ , and  $c_{i+1}(o) := 0$  if  $o \in L_i$ .
- Step 7. **Increase heuristic value:**  $h := h + 1$ .
- Step 8. Set  $i := i + 1$  and go to Step 1.

# Example



Adaptation/simplification of running example from Chapter 8:  
planning task  $\langle A, l, \{o_s, o_1, o_2, o_3, o_4, o_t\}, \gamma \rangle$  with

$$A = \{\mathbf{s}, a, b, c, d, e, f, g, h, \mathbf{t}\}$$

$$l = \{\mathbf{s} \mapsto 1, a \mapsto 0, b \mapsto 0, c \mapsto 0, d \mapsto 0, \\ e \mapsto 0, f \mapsto 0, g \mapsto 0, h \mapsto 0, \mathbf{t} \mapsto 0\}$$

$$o_s = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

$$o_1 = \langle c \wedge d, b \rangle$$

$$o_2 = \langle a \wedge b, e \rangle$$

$$o_3 = \langle a, f \rangle$$

$$o_4 = \langle f, g \wedge h \rangle$$

$$o_t = \langle e \wedge g \wedge h, \mathbf{t} \rangle$$

$$\gamma = \mathbf{t}$$

The LM-cut  
heuristic

Motivation

Definitions

Finding and  
exploiting  
landmarks

Admissibility

Summary



- Cheapest sequential (relaxed) plan:  $\langle o_s, o_1, o_2, o_3, o_4, o_t \rangle$   
with cost  $h^+(I) = 4$  (recall that  $o_s$  and  $o_t$  cost nothing).
- Parallel (relaxed) plan witnessing  $h_{\max}(I) = 2$ :  
 $\langle \{o_s\}, \{o_1, o_3\}, \{o_2, o_4\}, \{o_t\} \rangle$ .

**Our aim:** Get closer to  $h^+(I) = 4$  using  $h_{\text{LM-cut}}$  than using  $h_{\max}$ .



# Example: Iteration 1



prop $p$	<b>s</b>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<b>t</b>
$h_{\max}^{c_1}(p)$	0	0	1	0	0	2	1	2	2	2

action $o$	$o_s$	$o_1$	$o_2$	$o_3$	$o_4$	$o_t$
pcf $D_1(o)$	<b>s</b>	<i>c</i>	<i>b</i>	<i>a</i>	<i>f</i>	<i>g</i>

$$o_s[0] = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

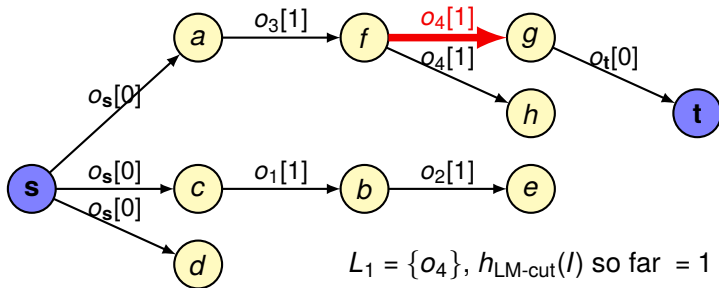
$$o_1[1] = \langle c \wedge d, b \rangle$$

$$o_2[1] = \langle a \wedge b, e \rangle$$

$$o_3[1] = \langle a, f \rangle$$

$$o_4[1] = \langle f, g \wedge h \rangle$$

$$o_t[0] = \langle e \wedge g \wedge h, \mathbf{t} \rangle$$



# Example: Iteration 2



prop $p$	<b>s</b>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<b>t</b>
$h_{\max}^{c_2}(p)$	0	0	1	0	0	2	1	1	1	2

action $o$	$o_s$	$o_1$	$o_2$	$o_3$	$o_4$	$o_t$
pcf $D_2(o)$	<b>s</b>	<i>c</i>	<i>b</i>	<i>a</i>	<i>f</i>	<i>e</i>

$$o_s[0] = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

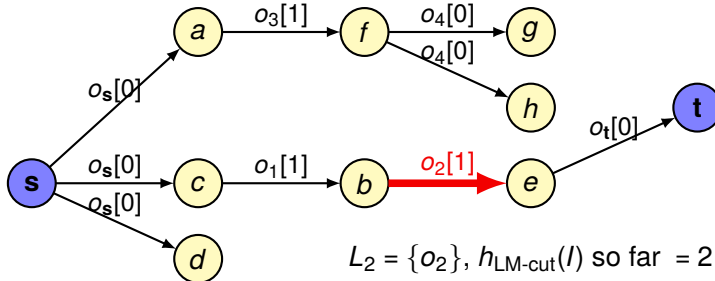
$$o_1[1] = \langle c \wedge d, b \rangle$$

$$o_2[1] = \langle a \wedge b, e \rangle$$

$$o_3[1] = \langle a, f \rangle$$

$$o_4[0] = \langle f, g \wedge h \rangle$$

$$o_t[0] = \langle e \wedge g \wedge h, t \rangle$$



# Example: Iteration 3



prop $p$	<b>s</b>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<b>t</b>
$h_{\max}^{c_3}(p)$	0	0	1	0	0	1	1	1	1	1

action $o$	$o_s$	$o_1$	$o_2$	$o_3$	$o_4$	$o_t$
pcf $D_3(o)$	<b>s</b>	<i>c</i>	<i>b</i>	<i>a</i>	<i>f</i>	<i>g</i>

$$o_s[0] = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

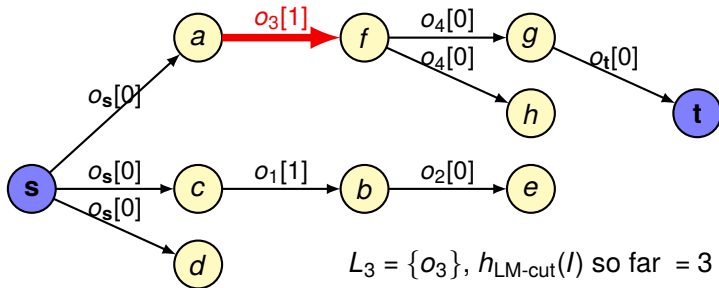
$$o_1[1] = \langle c \wedge d, b \rangle$$

$$o_2[0] = \langle a \wedge b, e \rangle$$

$$o_3[1] = \langle a, f \rangle$$

$$o_4[0] = \langle f, g \wedge h \rangle$$

$$o_t[0] = \langle e \wedge g \wedge h, \mathbf{t} \rangle$$



# Example: Iteration 4



prop $p$	<b>s</b>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<b>t</b>
$h_{\max}^4(p)$	0	0	1	0	0	1	0	0	0	1

action $o$	$o_s$	$o_1$	$o_2$	$o_3$	$o_4$	$o_t$
pcf $D_4(o)$	<b>s</b>	<i>c</i>	<i>b</i>	<i>a</i>	<i>f</i>	<i>e</i>

$$o_s[0] = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

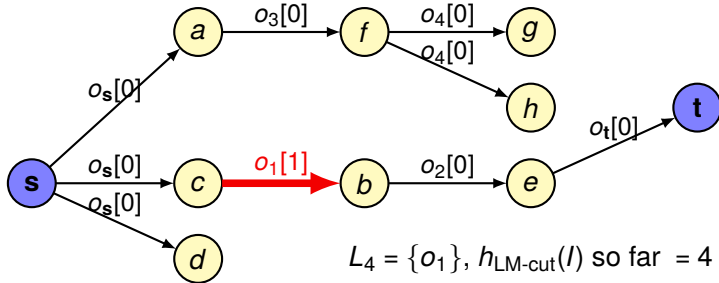
$$o_1[1] = \langle c \wedge d, b \rangle$$

$$o_2[0] = \langle a \wedge b, e \rangle$$

$$o_3[0] = \langle a, f \rangle$$

$$o_4[0] = \langle f, g \wedge h \rangle$$

$$o_t[0] = \langle e \wedge g \wedge h, t \rangle$$



# Example: Iteration 5



prop $p$	<b>s</b>	<i>a</i>	<i>b</i>	<i>c</i>	<i>d</i>	<i>e</i>	<i>f</i>	<i>g</i>	<i>h</i>	<b>t</b>
$h_{\max}^{c_5}(p)$	0	0	0	0	0	0	0	0	0	0

action $o$	$o_s$	$o_1$	$o_2$	$o_3$	$o_4$	$o_t$
pcf $D_5(o)$	<b>s</b>	<i>c</i>	<i>b</i>	<i>a</i>	<i>f</i>	<i>g</i>

$$o_s[0] = \langle \mathbf{s}, a \wedge c \wedge d \rangle$$

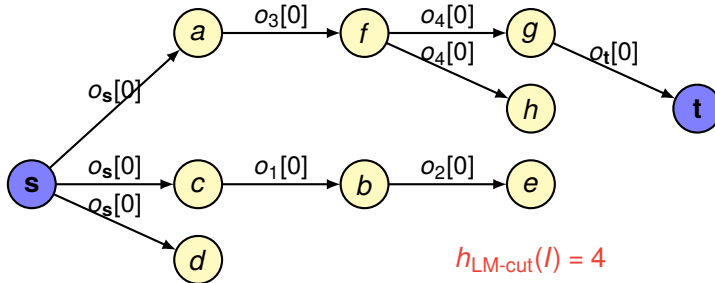
$$o_1[0] = \langle c \wedge d, b \rangle$$

$$o_2[0] = \langle a \wedge b, e \rangle$$

$$o_3[0] = \langle a, f \rangle$$

$$o_4[0] = \langle f, g \wedge h \rangle$$

$$o_t[0] = \langle e \wedge g \wedge h, \mathbf{t} \rangle$$





## Theorem

*The LM-cut heuristic never overestimates  $h^+$ , i.e., it is admissible.*

## Proof sketch

- From every landmark found, at least one operator has to be applied in any relaxed plan.
- Each found landmark is counted only once and there is no overlap in operators used in landmarks, i.e., the **landmarks** that are found **are disjoint** (operator costs for all operators in a “used” landmark are reset to zero).
- Therefore, we count at most as many landmarks as there are operators in a shortest relaxed plan.

The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

Admissibility

Summary



- **Remark:**  $h_{\text{LM-cut}}$  can be generalized to planning tasks with **non-unit costs**.
  - Instead of setting operator costs to zero, **decrease costs** of all operators in landmark by the minimal cost of any operator in the landmark.  
This effectively leads to a **cost partitioning** of operator costs between landmarks: An operator can be (partly) counted in more than one landmark, but the sum of the weights it is counted with will not exceed its true cost.
  - Instead of incrementing heuristic value by one in each step, increase it by minimal cost of any operator in the landmark.

Then,  $h_{\text{LM-cut}}$  is **still admissible**. Proof via cost-partitioning argument.

The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

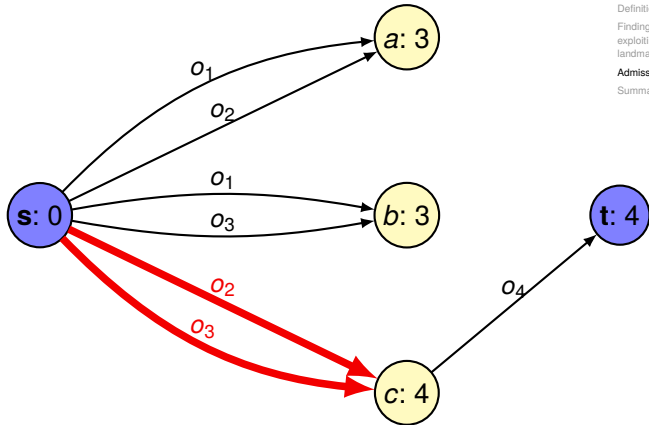
**Admissibility**

Summary

## Example

Iter. 1:  $D(\mathbf{t}) = a \rightsquigarrow L_1 = \{o_2, o_3\}$  [4]

$o_1[3] = \langle \mathbf{s}, a \wedge b \rangle$   
 $o_2[4] = \langle \mathbf{s}, a \wedge c \rangle$   
 $o_3[5] = \langle \mathbf{s}, b \wedge c \rangle$   
 $o_4[0] = \langle a \wedge b \wedge c, \mathbf{t} \rangle$



The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

Admissibility

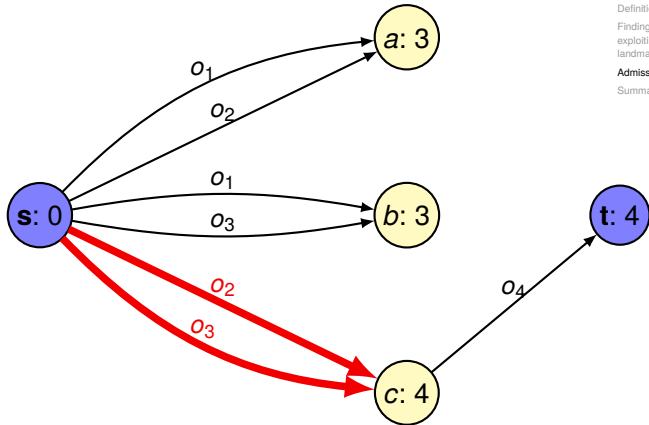
Summary



## Example

Iter. 1:  $D(\mathbf{t}) = a \rightsquigarrow L_1 = \{o_2, o_3\} [4] \rightsquigarrow h_{\text{LM-cut}}(l) := 4$

$o_1[3] = \langle \mathbf{s}, a \wedge b \rangle$   
 $o_2[0] = \langle \mathbf{s}, a \wedge c \rangle$   
 $o_3[1] = \langle \mathbf{s}, b \wedge c \rangle$   
 $o_4[0] = \langle a \wedge b \wedge c, \mathbf{t} \rangle$



The LM-cut heuristic

Motivation

Definitions

Finding and exploiting landmarks

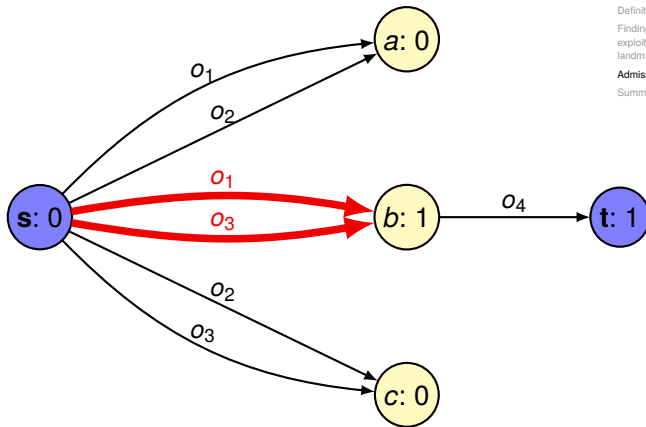
Admissibility

Summary

## Example

Iter. 2:  $D(\mathbf{t}) = b \rightsquigarrow L_2 = \{o_1, o_3\} [1]$

$o_1[3] = \langle \mathbf{s}, a \wedge b \rangle$   
 $o_2[0] = \langle \mathbf{s}, a \wedge c \rangle$   
 $o_3[1] = \langle \mathbf{s}, b \wedge c \rangle$   
 $o_4[0] = \langle a \wedge b \wedge c, \mathbf{t} \rangle$



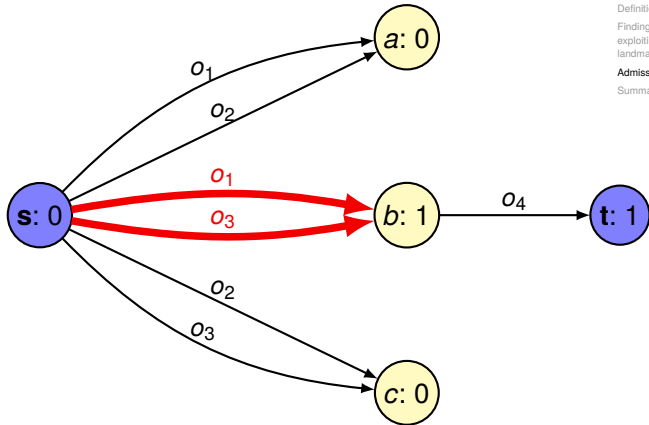
## The LM-cut heuristic

- Motivation
- Definitions
- Finding and exploiting landmarks
- Admissibility
- Summary

## Example

Iter. 2:  $D(\mathbf{t}) = b \rightsquigarrow L_2 = \{o_1, o_3\} [1] \rightsquigarrow h_{\text{LM-cut}}(l) := 4 + 1 = 5$

$o_1[2] = \langle \mathbf{s}, a \wedge b \rangle$   
 $o_2[0] = \langle \mathbf{s}, a \wedge c \rangle$   
 $o_3[0] = \langle \mathbf{s}, b \wedge c \rangle$   
 $o_4[0] = \langle a \wedge b \wedge c, \mathbf{t} \rangle$



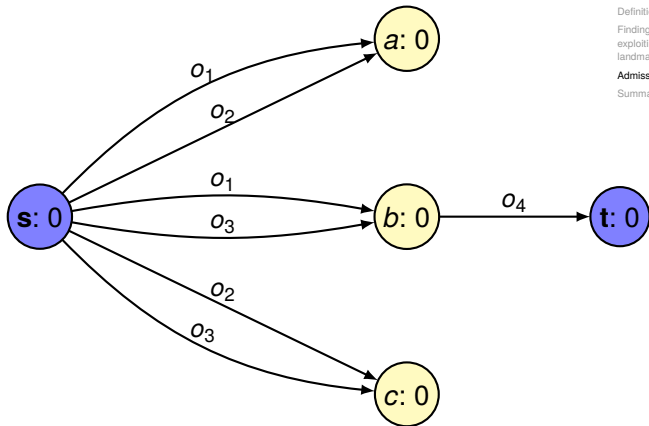
The LM-cut heuristic

- Motivation
- Definitions
- Finding and exploiting landmarks
- Admissibility
- Summary

## Example

Iter. 3:  $h_{\max}(\mathbf{t}) = 0 \rightsquigarrow$  done!  $\rightsquigarrow h_{\text{LM-cut}}(l) = 5$

$o_1[2] = \langle \mathbf{s}, a \wedge b \rangle$   
 $o_2[0] = \langle \mathbf{s}, a \wedge c \rangle$   
 $o_3[0] = \langle \mathbf{s}, b \wedge c \rangle$   
 $o_4[0] = \langle a \wedge b \wedge c, \mathbf{t} \rangle$



The LM-cut heuristic

- Motivation
- Definitions
- Finding and exploiting landmarks
- Admissibility
- Summary



**Remark:** The costs of  $o_3$  (i.e., 5) were **partitioned** as follows:

- 4 cost units were used in the cost of  $L_1$ , and
- 1 cost unit was used in the cost of  $L_2$ .

Without this cost partitioning, we would have only found  $L_1$  and counted it at a cost of 4. Landmark  $L_2$  would not have been considered, since it is **not** disjoint from  $L_1$ .

Thus, we would have arrived at an unnecessarily low value  $h_{\text{LM-cut}}(I) = 4$  instead of  $h_{\text{LM-cut}}(I) = 5$ .



- **Landmarks** are sets of actions such that each plan contains at least one of these actions.
- **Cuts** in **justification graphs** are a very general method to find landmarks.
- The **LM-cut heuristic** is an efficient admissible heuristic based on landmarks and cuts.
- It combines **delete relaxation**, **landmarks**, and **cost partitioning**.

## The LM-cut heuristic

- Motivation
- Definitions
- Finding and exploiting landmarks
- Admissibility
- Summary