Landmarks

Definitions, Discovery Methods and Uses

Erez Karpas[1] Silvia Richter[2 3]

[1]Faculty of Industrial Engineering and Management, Technion, Israel

[2]Griffith University, Queensland, Australia

[3]NICTA, Queensland, Australia

ICAPS 2010 Tutorial

Outline

SAS$^+$

- SAS$^+$ is a language for describing planning tasks compactly
- A SAS$^+$ task is given by a 4-tuple $\Pi = \langle \mathscr{V}, \mathscr{A}, s_0, G \rangle$
  - $\mathscr{V} = \{v_1, \ldots, v_n\}$ is a set of state variables, each associated with a finite domain $dom(v_i)$. By $v \mapsto d$ we denote that variable v is assigned value $d \in dom(v)$
  - Each complete assignment s to $\mathscr{V}$ is called a *state*
  - s_0 is an *initial state*
  - *goal* G is a partial assignment to $\mathscr{V}$
  - $\mathscr{A}$ is a finite set of *actions*
  - Each action a is a pair $\langle \text{pre}(a), \text{eff}(a) \rangle$ of partial assignments to $\mathscr{V}$ called *preconditions* and *effects*
- In cost-sensitive planning, each action a is also associated with a cost $C(a)$

Outline

1. What Landmarks Are
2. How Landmarks Are Discovered
3. Landmark Uses
   - Subgoals
   - Heuristic Estimates
   - Admissible Heuristic Estimates
   - Enriching the Problem
   - Beyond Classical Planning
4. Summary

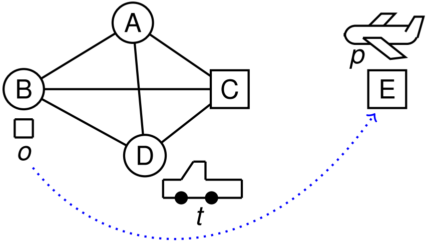
Landmarks

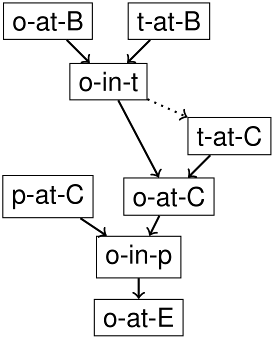
- A landmark is a formula that must be true at some point in every plan (Hoffmann, Porteous & Sebastia 2004)
- Landmarks can be (partially) ordered according to the order in which they must be achieved
- Some landmarks and orderings can be discovered automatically
- Current approaches consider only landmarks that are facts or disjunctions of facts
  (work on conjunctive landmarks to appear)

Action Landmarks

- An action landmark is an action which occurs in every plan
- Landmarks may imply actions landmarks (e.g., sole achievers)
- Action landmarks imply landmarks (e.g., preconditions and effects)
- Action landmarks might capture even more information
- Some action landmarks can be discovered automatically

Example Planning Problem - Logistics





Partial landmarks graph

Types of Landmark Orderings

- Sound landmark orderings are guaranteed to hold - they do not prune the solution space
- Unsound landmark orderings are additional constraints on plans - they may prune the solution space
- It is even possible that no plan exists that respects the unsound orderings
- However, unsound orderings are likely to hold and may save effort in planning

Sound Landmark Orderings

- *Natural* ordering $A \rightarrow B$, iff A true some time before B
- *Necessary* ordering $A \rightarrow_n B$, iff A always true before B becomes true
- *Greedy-necessary* ordering $A \rightarrow_{gn} B$, iff A true one step before B becomes true for the first time

Note that $A \rightarrow_n B \implies A \rightarrow_{gn} B \implies A \rightarrow B$

Reasonable Orderings

- Not sound - not guaranteed to hold in all plans
- *Reasonable* ordering $A \rightarrow_r B$, iff given B was achieved before A, any plan must delete B on the way to A, and re-achieve achieve B after or at the same time as A

$$B \rightsquigarrow \neg B \rightsquigarrow A \rightsquigarrow B \implies A \rightarrow_r B$$

- Initial state landmarks can be reasonably ordered after other landmarks (e. g., if they must be made false and true again)
- This can never happen with sound orderings

Obedient-Reasonable Orderings

- Assume that a plan obeys reasonable orderings $\Rightarrow$ find more orderings
- *Obedient-reasonable* ordering $A \rightarrow_{or} B$, iff given B was achieved before A, any plan that obeys reasonable orderings must delete B on the way to A and re-achieve B after or at the same time as A
- Not sound

Landmark Complexity

- Everything is PSPACE-complete

- Deciding if a given fact is a landmark is PSPACE-complete
- Proof Sketch: it's the same as deciding if the problem without actions that achieve this fact is unsolvable

- Deciding if there is a natural / necessary / greedy-necessary / reasonable ordering between two landmarks is PSPACE-complete

Outline

1. What Landmarks Are
2. How Landmarks Are Discovered
3. Landmark Uses
   - Subgoals
   - Heuristic Estimates
   - Admissible Heuristic Estimates
   - Enriching the Problem
   - Beyond Classical Planning
4. Summary

Landmark Discovery in Theory

Theory

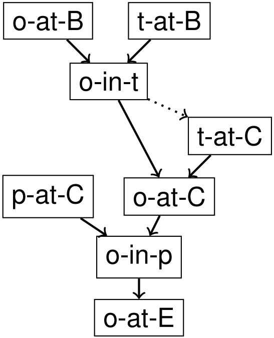
- A is a landmark $\iff \pi'_A$ is unsolvable
  where π'_A is π without the actions that achieve A

- The delete relaxation of π'_A is unsolvable $\Longrightarrow \pi'_A$ is unsolvable (delete-relaxation landmarks) – but better methods exist
- Other heuristics can be used to prove π'_A unsolvable: h^m, structural patterns, ...

Landmark Discovery I

Find landmarks and orderings by backchaining (Hoffmann et al. 2004, Porteous & Cresswell 2002)

- Every goal is a landmark
- If B is landmark and all actions that achieve B share A as precondition, then
  - A is a landmark
  - $A \rightarrow_n B$

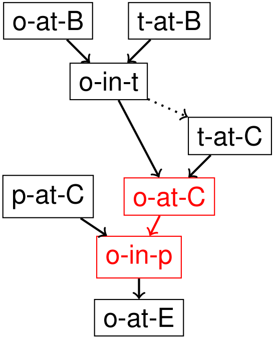


Useful restriction: consider only the case where B is achieved for the first time $\rightsquigarrow$ find more landmarks (and $A \rightarrow_{\mathrm{gn}} B$)

Landmark Discovery I

Find landmarks and orderings by backchaining (Hoffmann et al. 2004, Porteous & Cresswell 2002)

- Every goal is a landmark
- If B is landmark and all actions that achieve B share A as precondition, then
  - A is a landmark
  - $A \rightarrow_n B$

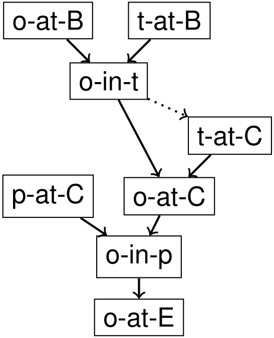


Useful restriction: consider only the case where B is achieved for the first time $\rightsquigarrow$ find more landmarks (and $A \rightarrow_{\mathrm{gn}} B$)

Landmark Discovery I (ctd.)

PSPACE-complete to find first achievers
$\leadsto$ over-approximation by building relaxed planning graph for π'_B

- This graph contains no actions that add B
- Any action applicable in this graph can possibly be executed before B first becomes true $\leadsto$ possible first achievers

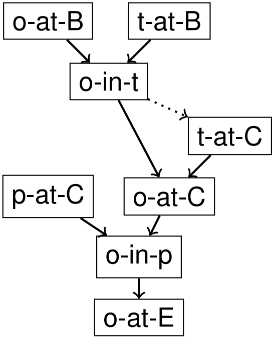


Additionally, if C not in the graph and C later proven to be a landmark, introduce $B \rightarrow C$

Landmark Discovery I (ctd.)

Disjunctive landmarks also possible,
e.g., (o-in-p_1 $\vee$ o-in-p_2):

- If B is landmark and all actions that (first) achieve B have A or C as precondition, then $A \vee C$ is a landmark
- Generalises to any number of disjuncts
- Large number of possible disjunctive landmarks $\rightsquigarrow$ must be restricted



Domain Transition Graphs (DTGs)

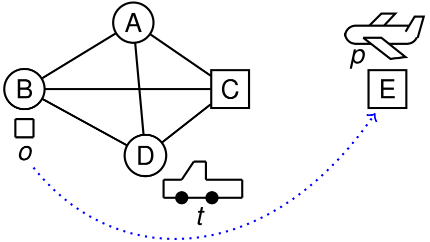
Find landmarks through DTGs (Richter et al. 2008)

The domain transition graph of $v \in \mathscr{V}$ (DTG_v) represents how the value of v can change.

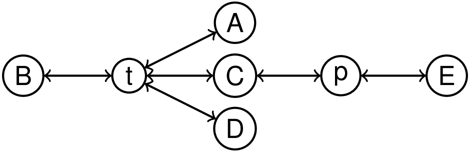
Given: a SAS^+ task $\langle \mathscr{V}, \mathscr{A}, s_0, G \rangle$
DTG_v is a directed graph with nodes $\mathscr{D}_v$ that has arc $\langle d, d' \rangle$ iff

- $d \neq d'$, and
- $\exists$ action with $v \mapsto d'$ as effect, and either
  - $v \mapsto d$ as precondition, or
  - no precondition on v

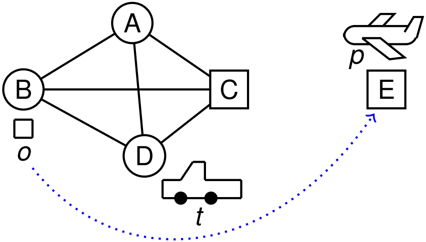
DTG Example



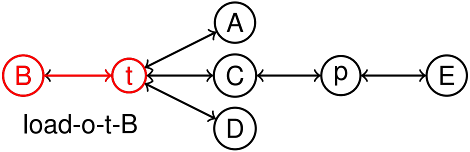
DTG for v_o:



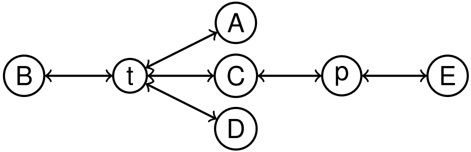
DTG Example



DTG for v_o:



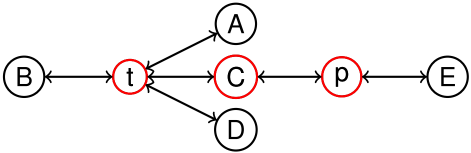
Landmark Discovery II



- Find landmarks through DTGs: if
  - $s_0(v) = d_0$,
  - $v \mapsto d$ landmark, and
  - every path from d_0 to d passes through d',

then $v \mapsto d'$ landmark, and $(v \mapsto d') \rightarrow (v \mapsto d)$

Landmark Discovery II



- Find landmarks through DTGs: if
  - $s_0(v) = d_0$,
  - $v \mapsto d$ landmark, and
  - every path from d_0 to d passes through d',

then $v \mapsto d'$ landmark, and $(v \mapsto d') \rightarrow (v \mapsto d)$

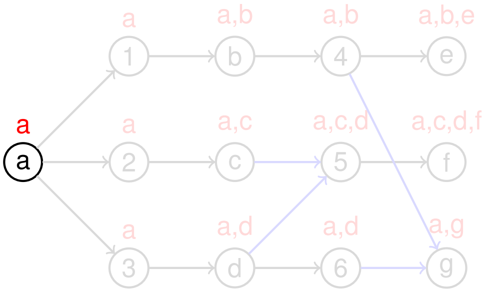
Landmark Discovery III

Find landmarks through forward propagation in relaxed planning graph (Zhu & Givan 2003)

- Finds causal landmarks only (preconditions for actions)
- Finds all causal delete-relaxation landmarks in polynomial time
- Propagate information on necessary predecessors
    - Label each fact node with itself
    - Propagate labels along arcs

Landmark Discovery III (ctd.)

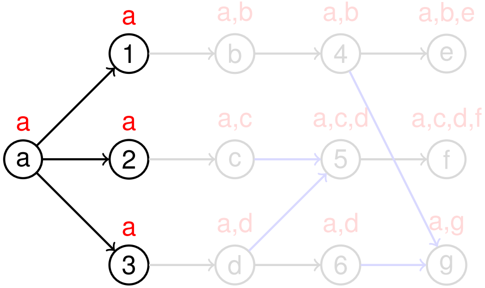
Actions (numbers): propagate union over labels on preconditions
— all preconditions are necessary
Facts (letters): propagate intersection over labels on achievers
— only what's necessary for all achievers is necessary for a fact



No-ops and repeated nodes not shown

Landmark Discovery III (ctd.)

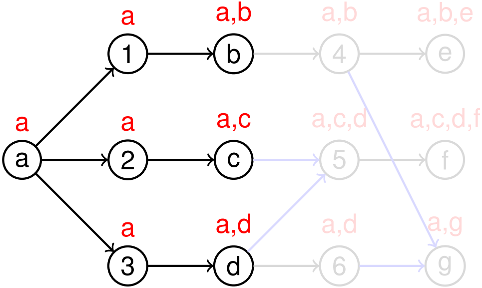
Actions (numbers): propagate union over labels on preconditions
— all preconditions are necessary
Facts (letters): propagate intersection over labels on achievers
— only what's necessary for all achievers is necessary for a fact



No-ops and repeated nodes not shown

Landmark Discovery III (ctd.)

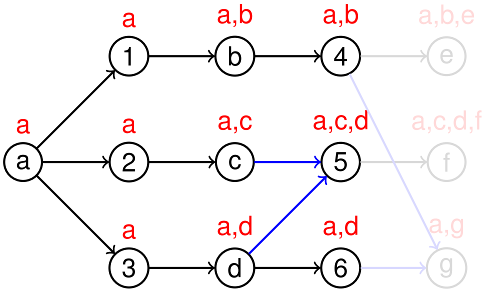
Actions (numbers): propagate union over labels on preconditions
— all preconditions are necessary
Facts (letters): propagate intersection over labels on achievers
— only what's necessary for all achievers is necessary for a fact



No-ops and repeated nodes not shown

Landmark Discovery III (ctd.)

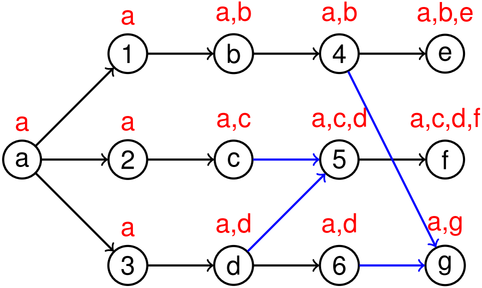
Actions (numbers): propagate union over labels on preconditions
— all preconditions are necessary
Facts (letters): propagate intersection over labels on achievers
— only what's necessary for all achievers is necessary for a fact



No-ops and repeated nodes not shown

Landmark Discovery III (ctd.)

Actions (numbers): propagate union over labels on preconditions
— all preconditions are necessary
Facts (letters): propagate intersection over labels on achievers
— only what's necessary for all achievers is necessary for a fact



No-ops and repeated nodes not shown

Landmark Discovery III (ctd.)

- Goal nodes in final layer: labels are landmarks
- $A \longrightarrow B$ if A forms part of the label for B in the final layer
- $A \longrightarrow_{\text{gn}} B$ if A is precondition for all possible first achievers of B
- Possible first achievers of B are achievers that do not have B in their label (Keyder, Richter & Helmert 2010)

Advanced version of this method counts re-occurrences of landmarks

Approximating reasonable orderings

We want to introduce $A \rightarrow_{\mathrm{r}} B$ if

1. B must be true after or at the same time as A's first occurrence, and
2. Achieving B first means losing it on the way to A

Approximating reasonable orderings (ctd.)

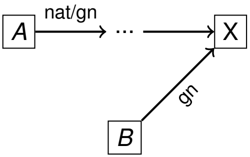
1. *B* must be true after or at the same time as *A*'s first occurrence

   Holds if

   - *B* is a goal

     or
   - there is a chain of natural/greedy-nec. orderings

     $A \rightarrow \ldots \rightarrow X$, and $B \rightarrow_{\mathrm{gn}} X$



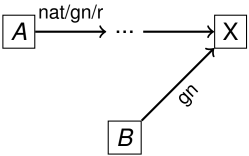
Approximating obedient reasonable orderings

1. B must be true after or at the same time as A's first occurrence

   Holds if

   - B is a goal

     or
   - there is a chain of natural/greedy-nec./reasonable orderings $A \rightarrow \ldots \rightarrow X$, and $B \rightarrow_{\text{gn}} X$



Approximating reasonable orderings (ctd.)

2. Achieving B first means losing it on the way to A

   Holds if

   - A and B are inconsistent (mutually exclusive), or
   - All actions achieving A have an effect inconsistent with B, or
   - There is a landmark X inconsistent with B and $X \rightarrow_{\mathrm{gn}} A$

Outline

1. What Landmarks Are
2. How Landmarks Are Discovered
3. Landmark Uses
   - Subgoals
   - Heuristic Estimates
   - Admissible Heuristic Estimates
   - Enriching the Problem
   - Beyond Classical Planning
4. Summary

Using Landmarks

- Some landmarks and orderings can be discovered efficiently
- So what can we do once we have these landmarks?

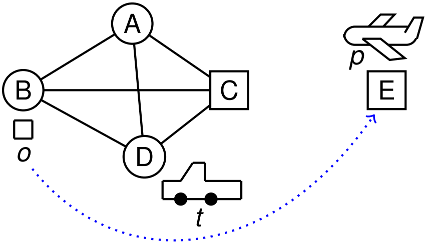
- We assume that landmarks and orderings are discovered in a pre-processing phase, and the same landmark graph is used throughout the planning phase

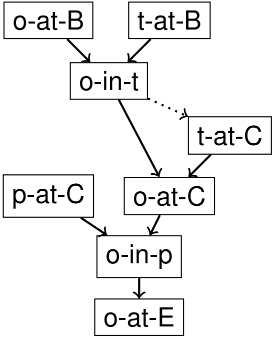
Outline

Using Landmarks as Subgoals

- Landmarks can be used as subgoals for a base planner
- The landmarks which could be achieved in the next iteration are passed as a disjunctive goal to a base planner
- After a landmark is achieved, repeat

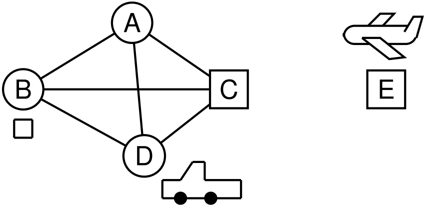
Using Landmarks as Subgoals - Logistics Example

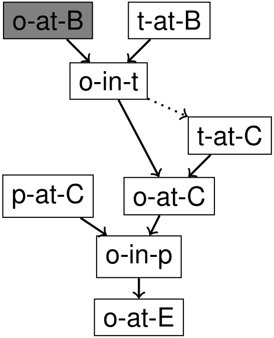




- Partial plan:
- Goal:

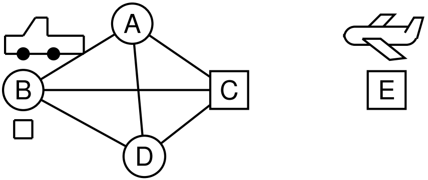
Using Landmarks as Subgoals - Logistics Example

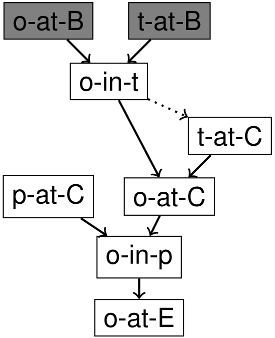




- Partial plan: $\emptyset$
- Goal: t-at-B $\vee$ p-at-C

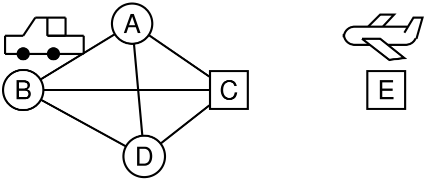
Using Landmarks as Subgoals - Logistics Example

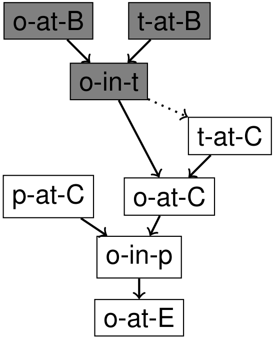




- Partial plan: Drive-t-B
- Goal: o-in-t ∨ p-at-C

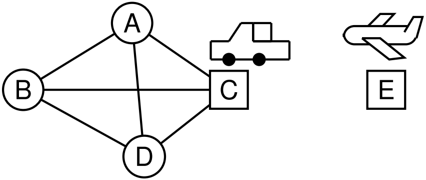
Using Landmarks as Subgoals - Logistics Example

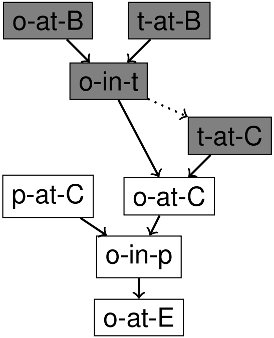




- Partial plan: Drive-t-B, Load-o-B
- Goal: t-at-C $\lor$ p-at-C

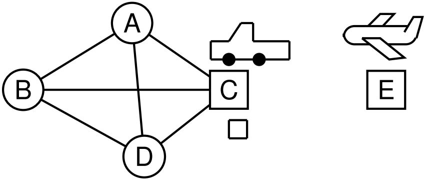
Using Landmarks as Subgoals - Logistics Example

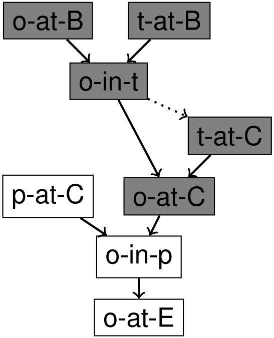




- Partial plan: Drive-t-B, Load-o-B, Drive-t-C
- Goal: o-at-C $\lor$ p-at-C

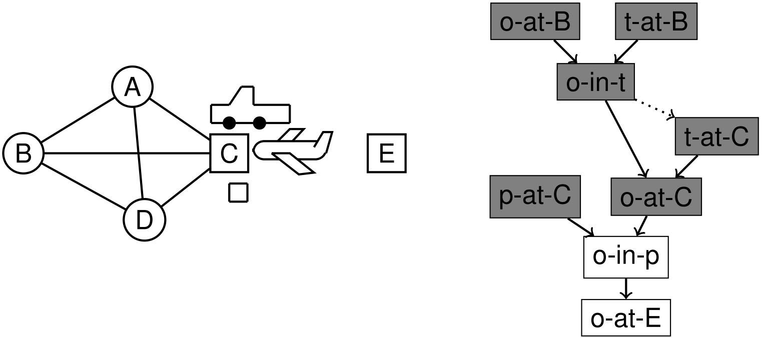
Using Landmarks as Subgoals - Logistics Example





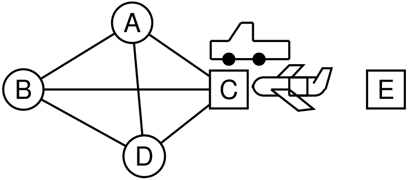
- Partial plan: Drive-t-B, Load-o-B, Drive-t-C, Unload-o-C
- Goal: p-at-C

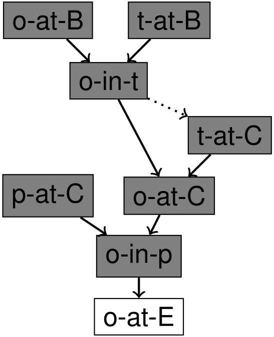
Using Landmarks as Subgoals - Logistics Example



- Partial plan: Drive-t-B, Load-o-B, Drive-t-C, Unload-o-C, Fly-p-C
- Goal: o-in-p

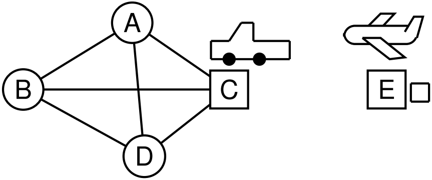
Using Landmarks as Subgoals - Logistics Example

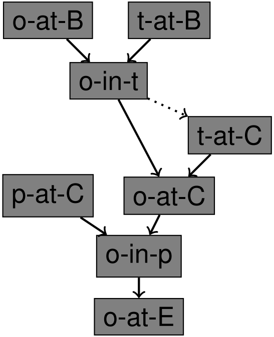




- Partial plan: Drive-t-B, Load-o-B, Drive-t-C, Unload-o-C, Fly-p-C, Load-o-p
- Goal: o-at-E

Using Landmarks as Subgoals - Logistics Example



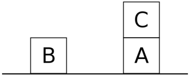


- Partial plan: Drive-t-B, Load-o-B, Drive-t-C, Unload-o-C, Fly-p-C, Load-o-p, Fly-p-E, Unload-o-E
- Goal: $\emptyset$

Using Landmarks as Subgoals

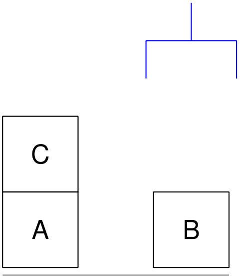
- That was a good example
- Now let's see a bad one

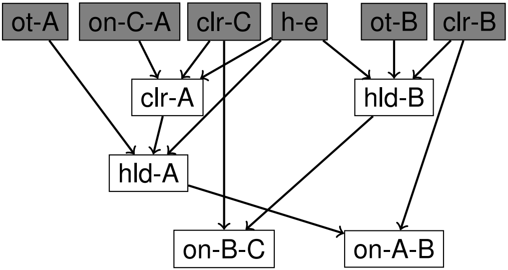
Using Landmarks as Subgoals - Sussman Example

- Consider the following blocks problem ("The Sussman Anomaly")
- Initial State 

- Goal: *on-A-B*, *on-B-C*

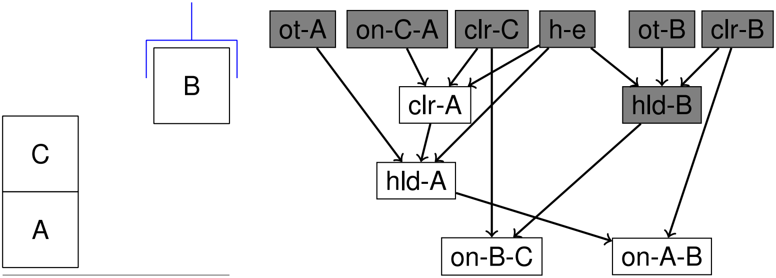
Using Landmarks as Subgoals - Sussman Example





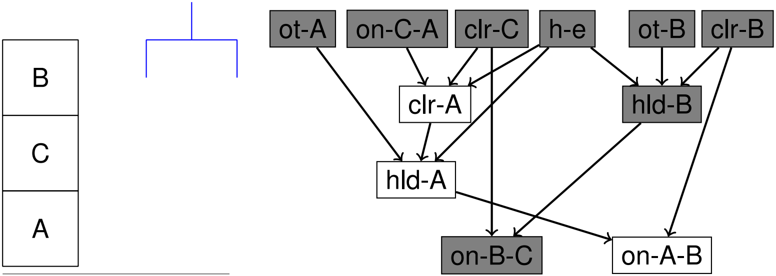
- Partial plan: $\emptyset$
- Goal: clear-A $\vee$ holding-B

Using Landmarks as Subgoals - Sussman Example



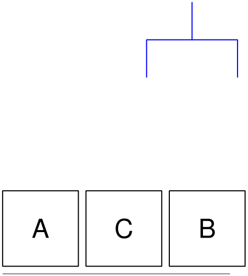
- Partial plan: Pickup-B
- Goal: clear-A $\vee$ on-B-C

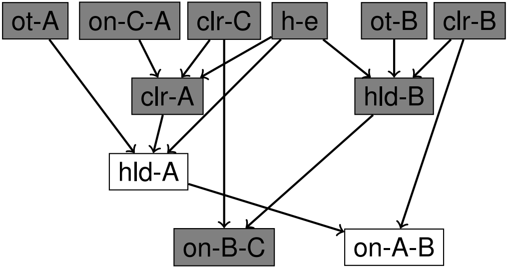
Using Landmarks as Subgoals - Sussman Example



- Partial plan: Pickup-B, Stack-B-C
- Goal: clear-A

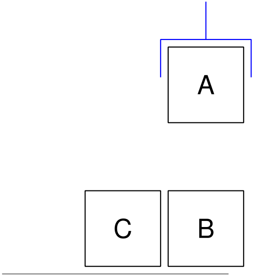
Using Landmarks as Subgoals - Sussman Example

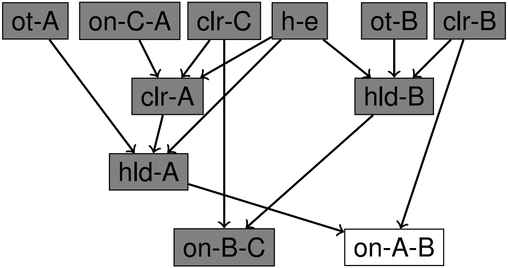




- Partial plan: Pickup-B, Stack-B-C, Unstack-B-C, Putdown-B, Unstack-C-A, Putdown-C
- Goal: holding-A

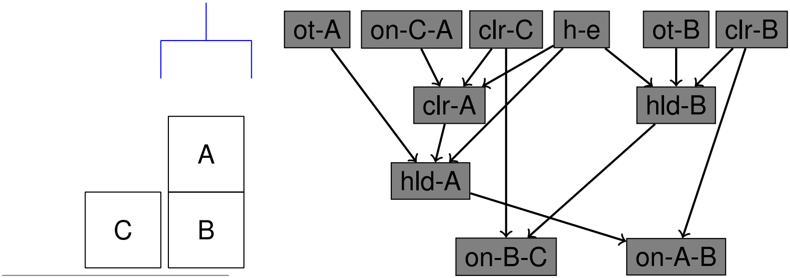
Using Landmarks as Subgoals - Sussman Example





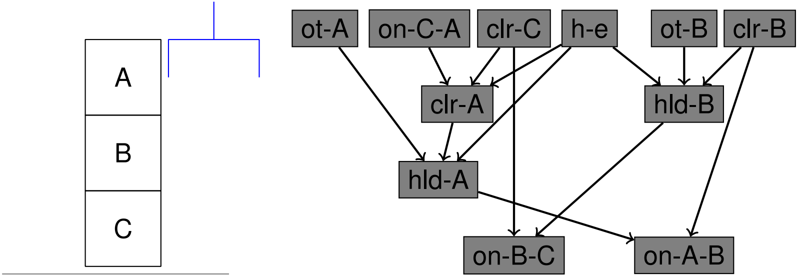
- Partial plan: Pickup-B, Stack-B-C, Unstack-B-C, Putdown-B, Unstack-C-A, Putdown-C, Pickup-A
- Goal: on-A-B

Using Landmarks as Subgoals - Sussman Example



- Partial plan: Pickup-B, Stack-B-C, Unstack-B-C, Putdown-B, Unstack-C-A, Putdown-C, Pickup-A, Stack-A-B
- Goal: Still need to achieve on-B-C

Using Landmarks as Subgoals - Sussman Example



- Partial plan: Pickup-B, Stack-B-C, Unstack-B-C, Putdown-B, Unstack-C-A, Putdown-C, Pickup-A, Stack-A-B, Unstack-A-B, Putdown-A, Pickup-B, Stack-B-C, Pickup-A, Stack-A-B
- Goal: $\emptyset$

Using Landmarks as Subgoals - Pros and Cons

- Pros:
  - Planning is very fast - the base planner needs to plan to a lesser depth
- Cons:
  - Can lead to much longer plans
  - Not complete in the presence of dead-ends

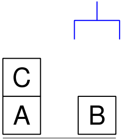
Outline

Using Landmarks for Heuristic Estimates

- The number of landmarks that still need to be achieved is a heuristic estimate (Richter, Helmert and Westphal 2008)
- Used by *LAMA* - winner of the IPC-2008 sequential satisficing track

Path-dependent Heuristics

- Suppose we are in state s. Did we achieve landmark A yet?
- Example: did we achieve holding(B)?



- There is no way to tell just by looking at s
- Achieved landmarks are a function of path, not state
- The number of landmarks that still need to be achieved is a path-dependent heuristic

The Landmark Heuristic

- The landmarks that still need to be achieved after reaching state s via path π are

$$L(s,\pi) = (L \setminus \mathrm{Accepted}(s,\pi)) \cup \mathrm{ReqAgain}(s,\pi)$$

- L is the set of all (discovered) landmarks
- $\mathrm{Accepted}(s,\pi) \subset L$ is the set of *accepted* landmarks
- $\mathrm{ReqAgain}(s,\pi) \subseteq \mathrm{Accepted}(s,\pi)$ is the set of *required again* landmarks - landmarks that must be achieved again

The Landmark Heuristic

- The landmarks that still need to be achieved after reaching state s via path π are

$$L(s,\pi) = (L \setminus \text{Accepted}(s,\pi)) \cup \text{ReqAgain}(s,\pi)$$

- L is the set of all (discovered) landmarks
- $\text{Accepted}(s,\pi) \subset L$ is the set of *accepted* landmarks
- $\text{ReqAgain}(s,\pi) \subseteq \text{Accepted}(s,\pi)$ is the set of *required again* landmarks - landmarks that must be achieved again

The Landmark Heuristic

- The landmarks that still need to be achieved after reaching state s via path π are

$$L(s,\pi) = (L \setminus \text{Accepted}(s,\pi)) \cup \text{ReqAgain}(s,\pi)$$

- L is the set of all (discovered) landmarks
- $\text{Accepted}(s,\pi) \subset L$ is the set of *accepted* landmarks
- $\text{ReqAgain}(s,\pi) \subseteq \text{Accepted}(s,\pi)$ is the set of *required again* landmarks - landmarks that must be achieved again

The Landmark Heuristic

- The landmarks that still need to be achieved after reaching state s via path π are

$$L(s,\pi) = (L \setminus \text{Accepted}(s,\pi)) \cup \text{ReqAgain}(s,\pi)$$

- L is the set of all (discovered) landmarks
- $\text{Accepted}(s,\pi) \subset L$ is the set of *accepted* landmarks
- $\text{ReqAgain}(s,\pi) \subseteq \text{Accepted}(s,\pi)$ is the set of *required again* landmarks - landmarks that must be achieved again

Accepted Landmarks

- In LAMA, a landmark A is first accepted by path π in state s if
  - all predecessors of A in the landmark graph have been accepted, and
  - A becomes true in s
- Once a landmark has been accepted, it remains accepted

Required Again Landmarks

- A landmark A is required again by path π in state s if:

  false-goal: A is false in s and is a goal, or

  open-prerequisite: A is false in s and is a greedy-necessary predecessor of some landmark B that is not accepted

- It's also possible to use (Buffet and Hoffmann, 2010):

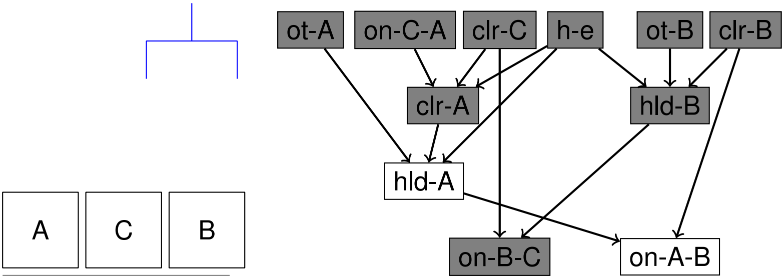
  doomed-goal: A is true in s and is a goal, but one of its greedy-necessary successors was not accepted, and is inconsistent with A

- Unsound rule:

  required-ancestor: is the transitive closure of *open-prerequisite*

Accepted and Required Again Landmarks - Example

- In the Sussman anomaly, after performing: Pickup-B, Stack-B-C, Unstack-B-C, Putdown-B, Unstack-C-A, Putdown-C



- on-B-C is a *false-goal*, and so it is required again

Problems With Reasonable Orderings

- Heuristic value of a goal state may be non-zero
  (if the plan found does not obey all reasonable orderings, and consequently not all landmarks are accepted).
  Solution: explicitely test states for goal condition
- The definition of reasonable orderings allows an ordering $A \rightarrow_r B$ if A and B become true simultaneously.
  Two solutions:
  - Accept a landmark if it has been made true at the same time as its predecessor (Buffet and Hoffmann, 2010)
  - Modify the definition of reasonable orderings to disallow such orderings

Outline

Admissible Heuristic Estimates

- LAMA's heuristic: the number of landmarks that still need to be achieved (Richter, Helmert and Westphal 2008)
- LAMA's heuristic is inadmissible - a single action can achieve multiple landmarks
  - Example: *hand-empty* and *on-A-B* can both be achieved by *stack-A-B*
- Admissible heuristic: assign a cost to each landmark, sum over the costs of landmarks (Karpas and Domshlak, 2009)

Conditions for Admissibility

- Each action shares its cost between all the landmarks it achieves

$$\forall a \in \mathscr{A} : \sum_{B \in L(a|s,\mathscr{P})} cost(a, B) \leq \mathrm{C}(a)$$

$cost(a, B)$: cost “assigned” by action a to B
$L(a|s, \mathscr{P})$: the set of landmarks achieved by a

- Each landmark is assigned at most the cheapest cost any action assigned it

$$\forall B \in L(s, \mathscr{P}) : \ cost(B) \leq \min_{a \in \mathrm{ach}(B|s,\mathscr{P})} cost(a, B)$$

$cost(B)$: cost assigned to landmark B
$\mathrm{ach}(B|s, \mathscr{P})$: the set of actions that can achieve B

Conditions for Admissibility

- Each action shares its cost between all the landmarks it achieves

$$\forall a \in \mathscr{A} : \sum_{B \in L(a|s,\mathscr{P})} cost(a,B) \leq \mathrm{C}(a)$$

$cost(a,B)$: cost "assigned" by action a to B
$L(a|s,\mathscr{P})$: the set of landmarks achieved by a

- Each landmark is assigned at most the cheapest cost any action assigned it

$$\forall B \in L(s,\mathscr{P}) : \ cost(B) \leq \min_{a \in \mathrm{ach}(B|s,\mathscr{P})} cost(a,B)$$

$cost(B)$: cost assigned to landmark B
$\mathrm{ach}(B|s,\mathscr{P})$: the set of actions that can achieve B

Admissible Cost Sharing

- Idea: the cost of a set of landmarks is no greater than the cost of any single action that achieves them
- Given that, the sum of costs of landmarks that still need to be achieved is an admissible heuristic, h_L

$$h_L(s,\pi) := cost(L(s,\pi)) = \sum_{B \in L(s,\pi)} cost(B)$$

- Proof: left up to the reader $\ddot\smile$

Cost Partitioning - how?

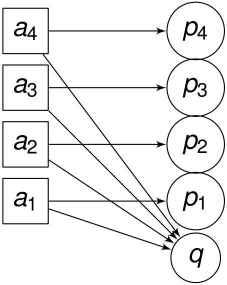
- How can we find such a partitioning?
- Easy answer - uniform cost sharing - each action shares its cost equally between the landmarks it achieves

$$cost(a, B) = \frac{C(a)}{|L(a|s, \pi)|}$$

$$cost(B) = \min_{a \in \text{ach}(B|s,\pi)} cost(a, B)$$

Uniform Cost Sharing

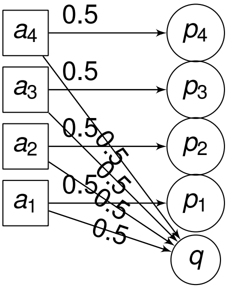
- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning



Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

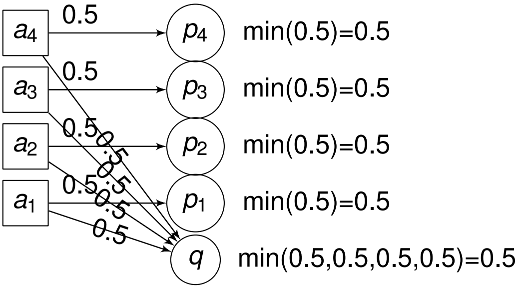
Uniform cost sharing



Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

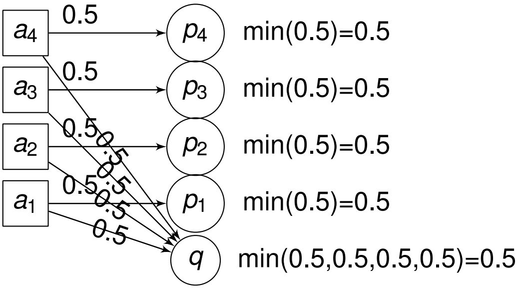
Uniform cost sharing



Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

Uniform cost sharing $h_L = 2.5$

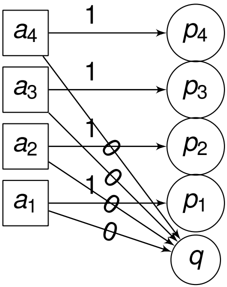


Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

Optimal cost sharing

uniform $h_L = 2.5$

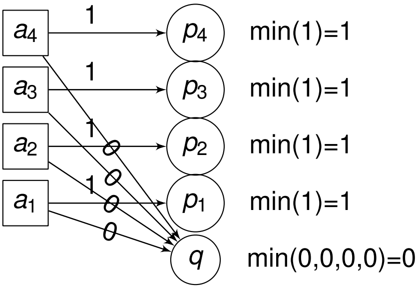


Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

Optimal cost sharing

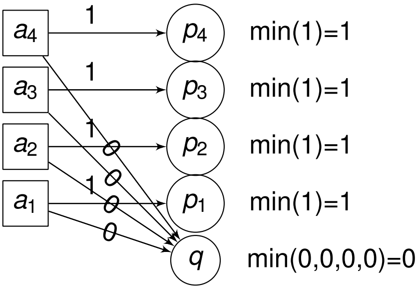
uniform $h_L = 2.5$



Uniform Cost Sharing

- Advantage: Easy and fast to compute
- Disadvantage: can be much worse than the optimal cost partitioning

Optimal cost sharing $h_L = 4$ uniform $h_L = 2.5$



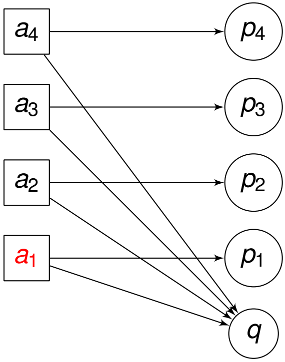
Optimal Cost Sharing

- The good news: the optimal cost partitioning is poly-time to compute
  - The constraints for admissibility are linear, and can be used in a Linear Program (LP)
  - Objective: maximize the sum of landmark costs
  - The solution to the LP gives us the optimal cost partitioning
- The bad news: poly-time can still take a long time

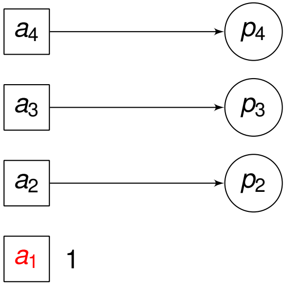
How can we do better?

- So far:
  - Uniform cost sharing is easy to compute, but suboptimal
  - Optimal cost sharing takes a long time to compute
- Q: How can we get better heuristic estimates that don't take a long time to compute?
- A: Exploit additional information - action landmarks

Using Action Landmarks - by Example



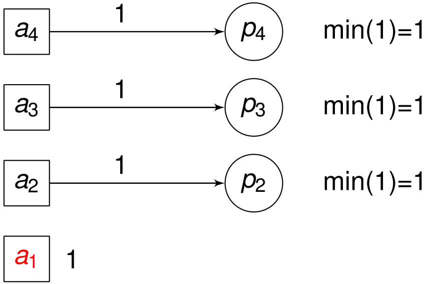
Using Action Landmarks - by Example



Using Action Landmarks - by Example

Uniform Cost Sharing

$h_{LA} = 4$



Outline

Using Landmarks to Enrich the Problem

- Landmarks are, in essence, implicit goals
- We can make these explicit by reformulating the planning problem
- Two different methods for doing this have been proposed (Wang, Baier and McIlraith, 2009 and Domshlak, Katz and Lefler, 2010)

Viewing Landmarks as Temporally Extended Goals

- Landmarks and their orderings can be viewed as temporally extended goals (Wang, Baier and McIlraith, 2009)
- These temporally extended goals can be expressed in Linear Temporal Logic (LTL)
- Each LTL formula can be compiled into a finite-state automaton
- Each FSA can be encoded as a single variable in an enriched planning problem

A Simpler Approach

- A simpler approach of encoding landmarks into a planning problem is to encode the landmarks directly (Domshlak, Katz and Lefler, 2010)
  - Each landmark is represented by a single binary state variable
  - The two values represent landmark accepted / not accepted
  - Each action that achieves the landmark has an additional effect added to it, changing the landmark variable value to accepted
  - The accepting value of each landmark variable is added to the goal state

Why Enrich Problems?

- Landmarks and orderings are implicit, encoding them into the problem makes them explicit
- Allows other heuristics to use landmark information
- Example: structural pattern heuristic on the enriched problem accounts not only for explicit goals (Domshlak, Katz and Lefler, 2010)
- In fact, the landmark count heuristic can be seen as the goal count heuristic on the landmark enriched problem
- Caveat - since current landmark discovery procedures are based on delete-relaxation, this adds no information to delete-relaxation based heuristics

Outline

Landmarks in Probabilistic Planning

- Landmarks can also be adapted to probabilistic planning (Buffet and Hoffmann, 2010)
- In a probabilistic planning task, a landmark is a fact which must be true in every successful trajectory (possible execution)
- Using the landmark count heuristic in FPG yields good results

Outline

Summary

- Landmarks provide a way to utilize the implicit structure of a planning problem
- They can (and have been) used successfully for both satisficing and optimal planning

Challenges

- Discover more landmarks and orderings using different techniques (why use only relaxed planning?)
- Discover and exploit more complex types of landmarks (conjunctive, CNF, first order logic . . .)
- Discover and exploit landmarks in a problem-independent manner (for example, in Logistics - a package that is not at its destination must always be loaded on a truck or an airplane)

More Challenges

- A landmark / ordering must be true in every plan
- An optimal landmark / ordering must be true in every optimal plan
  - All landmarks are also optimal landmarks, but can we find facts which are optimal landmarks and NOT landmarks?
- An existential landmark / ordering must be true in a plan
  - We must be careful not to mix between existential landmarks from different plans
- An existential optimal landmark / ordering must be true in an optimal plan

More Challenges

- A landmark / ordering must be true in every plan
- An optimal landmark / ordering must be true in every optimal plan
  - All landmarks are also optimal landmarks, but can we find facts which are optimal landmarks and NOT landmarks?
- An existential landmark / ordering must be true in a plan
  - We must be careful not to mix between existential landmarks from different plans
- An existential optimal landmark / ordering must be true in an optimal plan

More Challenges

- A landmark / ordering must be true in every plan
- An optimal landmark / ordering must be true in every optimal plan
  - All landmarks are also optimal landmarks, but can we find facts which are optimal landmarks and NOT landmarks?
- An existential landmark / ordering must be true in a plan
  - We must be careful not to mix between existential landmarks from different plans
- An existential optimal landmark / ordering must be true in an optimal plan

More Challenges

- A landmark / ordering must be true in every plan
- An optimal landmark / ordering must be true in every optimal plan
  - All landmarks are also optimal landmarks, but can we find facts which are optimal landmarks and NOT landmarks?
- An existential landmark / ordering must be true in a plan
  - We must be careful not to mix between existential landmarks from different plans
- An existential optimal landmark / ordering must be true in an optimal plan

Thank You

Enjoy ICAPS!

References

- Jörg Hoffmann, Julie Porteous and Laura Sebastia.
  Ordered landmarks in planning.
  JAIR 22, pp. 215–278, 2004.
  Introduces landmarks and the backchaining generation method.
  (Longer version of a 2001 article by the same authors.)
- Julie Porteous and Stephen Cresswell.
  Extending landmarks analysis to reason about resources and repetition.
  Proc. PLANSIG 02, pp. 45–54, 2002.
  Introduces the “possibly before” approximation using RPGs for first achievers of landmarks.

References (ctd.)

- Silvia Richter, Malte Helmert and Matthias Westphal.
  Landmarks revisited.
  Proc. AAAI 2008, pp. 975–982, 2008.
  Describes the DTG method for finding landmarks and the landmark heuristic used by LAMA.
- Lin Zhu and Robert Givan.
  Landmark extraction via planning graph propagation.
  Proc. ICAPS 2003 Doctoral Consortium, pp. 156–160, 2003.
  Describes the method for finding the complete set of causal delete-relaxation landmarks in polynomial time.

References (ctd.)

- Peter Gregory, Stephen Cresswell, Derek Long and Julie Porteous.
  On the Extraction of Disjunctive Landmarks from Planning Problems via Symmetry Reduction.
  Proc. SumCon 2004, pp. 34–41, 2004.
  Further methods for finding disjunctive landmarks (not discussed in this tutorial).
- Emil Keyder, Silvia Richter and Malte Helmert.
  Sound and Complete Landmarks for And/Or Graphs
  Proc. ECAI 2010, to appear.
  Conjunctive landmarks (not discussed here); explains how to approximate greedy-necessary orderings in Zhu & Givan's landmark discovery method.

References (ctd.)

- Erez Karpas and Carmel Domshlak.
  Cost-optimal Planning with Landmarks.
  Proc. IJCAI 2009, pp. 1728–1733, 2009.
  Describes the admissible landmark heuristic and multi-path dependence.
- Olivier Buffet and Jörg Hoffmann.
  All that Glitters is not Gold: Using Landmarks for Reward Shaping in FPG.
  Proc. ICAPS 2010 Workshop on Planning and Scheduling under Uncertainty
  Describes how landmarks can be used in probabilistic planning, and some enhancements to accepted/required again definitions.

References (ctd.)

- Letao Wang, Jorge A. Baier and Sheila A. McIlraith .
  Viewing Landmarks as Temporally Extended Goals.
  Proc. ICAPS 2009 Workshop on Heuristics for Domain Independent Planning, 2009.
  Describes how to compile landmarks and orderings into a planning problem via LTL.
- Carmel Domshlak, Michael Katz and Sagi Lefler.
  When Abstractions Met Landmarks.
  Proc. ICAPS 2010, 2010.
  Describes how to compile landmarks into a planning problem and use them with abstraction heuristics.