Week 7 T6
Wednesday, 31 August 2022 4:33 PM

Relaxed plan

Set of actions to reach goal state from current state in relaxed problem

Definition (Relaxed Plan Heuristic). *A heuristic function is called a relaxed plan heuristic, denoted h^{FF}, if, given a state s, it returns ∞ if no relaxed plan exists, and otherwise returns $\sum_{a \in RPlan} c(a)$ where $RPlan$ is the action set returned by relaxed plan extraction on a closed well-founded best-supporter function for s.*

→ Recall: If a relaxed plan exists, then there also exists a closed well-founded best-supporter function, see previous slide.

Best Supporter Functions

notation: $bs(p)$ = best supporter for fact p

Action that makes fact p true which has "lowest accumulated cost" in approximated relaxation problem.

i.e. action with lowest cost + lowest cost of achieving prerequisites

Can use h^{add} and h^{max} to find best supporters
bs^{add} bs^{max}

$bs(p)$ is only defined for actions that are not true in the initial state and not defined for facts that are unreachable from current state in relaxed problem

Definition (Best-Supporters from h^{max} and h^{add}). *Let $\Pi = (F, A, c, I, G)$ be a STRIPS planning task, and let s be a state.*
The h^{max} supporter function $bs_s^{max} : \{p \in F \mid 0 < h^{max}(s, \{p\}) < \infty\} \mapsto A$ is defined by $bs_s^{max}(p) := \arg\min_{a \in A, p \in add_a} c(a) + h^{max}(s, pre_a)$.
The h^{add} supporter function $bs_s^{add} : \{p \in F \mid 0 < h^{add}(s, \{p\}) < \infty\} \mapsto A$ is defined by $bs_s^{add}(p) := \arg\min_{a \in A, p \in add_a} c(a) + h^{add}(s, pre_a)$.

Finding Best supporters using h^{add} / h^{max}

While performing Bellman Ford, maintain array/dictionary for storing BS for each action. Initialise each BS to NONE

- In each iteration, apply every possible action given preconditions. If the action updates the value of a fact p (makes true or reduces value), set $BS(p)$ = action

Ideal Properties of $bs(p)$

Closed: bs defined for every fact p

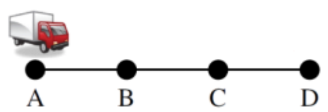
which may contribute to reaching goal
(except those already true in initial state)

Well-founded: paths using best supporters don't run into cycles.

Support Graph: Directed graph showing connections between facts and best supporter actions

Well founded $\Longleftrightarrow$ Support graph is acyclic

Support Graphs and Prerequisite (C) in "Logistics"



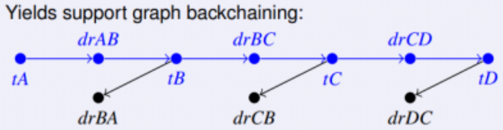
- Initial state: tA.
- Goal: tD.
- Actions: $drXY$.

How to do it (well-founded)

Best-supporter function:

p	$bs(p)$
$t(B)$	$dr(A,B)$
$t(C)$	$dr(B,C)$
$t(D)$	$dr(C,D)$

Yields support graph backchaining:

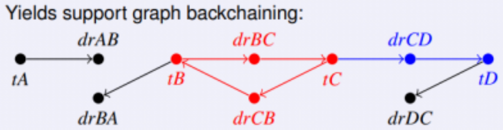


How to NOT do it (not well-founded)

Best-supporter function:

p	$bs(p)$
$t(B)$	$dr(C,B)$
$t(C)$	$dr(B,C)$
$t(D)$	$dr(C,D)$

Yields support graph backchaining:



Using h^{add} and h^{max} to derive best supporters guarantees closure and wellfoundedness

Calculating h^{FF}

Relaxed Plan Extraction for state s and best-supporter function bs

$Open := G \setminus s$; $Closed := \emptyset$; $RPlan := \emptyset$
while $Open \neq \emptyset$ **do**:
  select $g \in Open$
  $Open := Open \setminus \{g\}$; $Closed := Closed \cup \{g\}$;
  $RPlan := RPlan \cup \{bs(g)\}$; $Open := Open \cup (pre_{bs(g)} \setminus (s \cup Closed))$
endwhile
return $RPlan$

$\rightarrow$ Starting with the top-level goals, iteratively close open singleton sub-goals by selecting the best supporter.

This is fast! Number of iterations bounded by $|P|$, each near-constant time.

- Similar to BFS / DFS
- Initialise open set with all goal facts not true

in initial state, closed set with all initial facts, and empty action set

- While openset is not empty:

  Pop any element from openset. Add to closed set

  Add best supporter to action Set. Add all preconditions for action not in closed set to open set

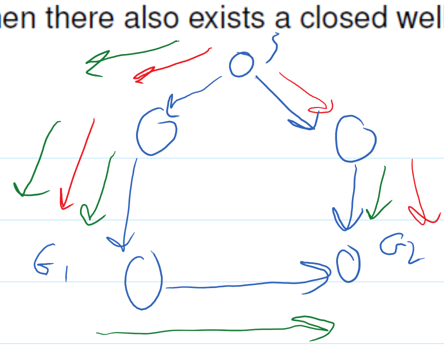
h^{ff} = number of actions in action set (unique actions)

Definition (Relaxed Plan Heuristic). *A heuristic function is called a relaxed plan heuristic, denoted h^{FF}, if, given a state s, it returns ∞ if no relaxed plan exists, and otherwise returns $\sum_{a \in RPlan} c(a)$ where $RPlan$ is the action set returned by relaxed plan extraction on a closed well-founded best-supporter function for s.*

→ Recall: If a relaxed plan exists, then there also exists a closed well-founded best-supporter function, see previous slide.

$h^{ff} \geq h^{+}$

- May be inadmissible
- $h^{ff}(s) = \infty \iff h^*(s) = \infty$



Definition (Helpful Actions) *Let h^{FF} be a relaxed plan heuristic, let s be a state, and let $RPlan$ be the action set returned by relaxed plan extraction on the closed well-founded best-supporter function for s which underlies h^{FF}. Then an action a applicable to s is called helpful if it is contained in $RPlan$.*

Remarks

- Initially introduced in FF [Hoffmann and Nebel (2011)], restricting Enforced Hill-Climbing to use **only** the helpful actions
- Expanding only helpful actions does not guarantee completeness.
- Other planners use helpful actions as prefered operators, expanding first nodes resulting from helpful actions.

WIDTH

Sort of Measure of Problem Difficulty

- low width = easy

  Single fact in goal set

  or if multiple goals, easy to achieve one at a time

- High width = hard

  Many many goal conditions

or goals are hard to achieve one at a time
(Solving one makes others harder)

Novelty

Key definition: the **novelty** $w(s)$ **of a state** s is the size of the smallest subset of atoms in s that is true for the first time in the search.

- e.g. $w(s) = 1$ if there is **one** atom $p \in s$ such that s is the first state that makes p true.
- Otherwise, $w(s) = 2$ if there are **two** different atoms $p, q \in s$ such that s is the first state that makes $p \wedge q$ true.
- Otherwise, $w(s) = 3$ if there are **three** different atoms...

Pseudocode:

- Initialise empty set to store seen facts (Seen set)
- While performing search algorithm:
  1) Add PowerSet of facts to seen fact set
     i.e if state = {A, B, C} then add:
     {A, B, C, (A,B), (A,C), (B,C), (A,B,C)}
  2) if new single fact added, then novelty = 1
  3) else if new pair of facts added, then novelty = 2
  4) else if new triple, novelty = 3 etc
  5) If nothing new added, novelty = 1 + num facts in state

Iterated Width (*IW*)

Algorithm

- $IW(k)$ = breadth-first search that prunes newly generated states whose $novelty(s) > k$.
- IW is a sequence of calls $IW(k)$ for $i = 0, 1, 2, \ldots$ over problem P until problem solved or i exceeds number of variables in problem

Properties

$IW(k)$ expands at most $O(n^k)$ states, where n is the number of atoms.

- For IW(k), don't need to calculate novelty if it is guaranteed to be bigger than k, can just prune

Dealing with multiple goal facts?

Use Serialised iterative width

- IW from initial state to first goal
- Then IW from first goal to next goal etc