

COMP90054 – Week 7 tutorial

Prepared by Zijie Xu. Last updated: 13 September 2023

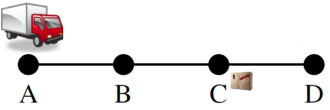
Best-Supporters and Relaxed plan heuristics

Best-supporter functions bs

Definition

- Best-supporter function bs : For every fact $p \in (F \setminus s)$, returns an action that can achieve p with the cheapest cost within relaxation
- In this subject we use the two following closed well-founded best-supporter functions
 - $bs_s^{\text{add}}(p) := \underset{a \in A, p \in \text{add}_a}{\text{argmin}} \left(c(a) + h^{\text{add}}(s, \text{pre}_a) \right)$
 - $bs_s^{\text{max}}(p) := \underset{a \in A, p \in \text{add}_a}{\text{argmin}} \left(c(a) + h^{\text{max}}(s, \text{pre}_a) \right)$

Example



- Initial state $I: t(A), p(C)$.
- Goal $G: t(A), p(D)$.
- Actions $A: dr(X, Y), lo(X), ul(X)$.

• “Logistics”:

	$t(A)$	$t(B)$	$t(C)$	$t(D)$	$p(T)$	$p(A)$	$p(B)$	$p(C)$	$p(D)$
h^{add}	0	1	2	3	3	4	5	0	7
bs_s^{add}	-	$dr(A, B)$	$dr(B, C)$	$dr(C, D)$	$lo(C)$	$ul(A)$	$ul(B)$	-	$ul(D)$

- For $p = t(B)$, $bs_s^{\text{add}}(p) = \underset{a \in \{dr(A, B), dr(C, B)\}}{\text{argmin}} \left(c(a) + h^{\text{add}}(s, \text{pre}_a) \right)$
 - $a = dr(A, B) \rightarrow 1 + h^{\text{add}}(t(A)) = 1$
 - $a = dr(C, B) \rightarrow 1 + h^{\text{add}}(t(C)) = 3$

Relaxed plan heuristics h^{FF}

Idea

1. Compute a best-supporter function bs
2. Extract a relaxed plan $RPlan$ by applying bs to each $g \in G$ and collecting all actions
3. Use the relaxed plan heuristics $h^{FF} = \sum_{a \in RPlan} c(a)$ to estimate h^*

Algorithm

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

```

Open := G \ s; Closed := ∅; RPlan := ∅
while Open ≠ ∅ do:
  select g ∈ Open
  Open := Open \ {g}; Closed := Closed ∪ {g};
  RPlan := RPlan ∪ {bs(g)}; Open := Open ∪ (prebs(g) \ (s ∪ Closed))
endwhile
return RPlan

```

- Requires a closed well-founded best-supporter function to make sense

Properties of h^{FF}

- h^{FF} is pessimistic: $h^{FF} \geq h^+$ so $\exists \Pi, s$ such that $h^{FF}(s) > h^*(s)$
- h^{FF} agrees with h^+ on ∞ : $h^{FF}(s) = \infty \Leftrightarrow h^+(s) = \infty$
- h^{FF} does not over-count like h^{add}

Example



A B C D

- Initial state $I: t(A), p(C)$.
- Goal $G: t(A), p(D)$.
- Actions $A: dr(X, Y), lo(X), ul(X)$.

- Logistics’:

	$t(A)$	$t(B)$	$t(C)$	$t(D)$	$p(T)$	$p(A)$	$p(B)$	$p(C)$	$p(D)$
bs_s^{add}	-	$dr(A, B)$	$dr(B, C)$	$dr(C, D)$	$lo(C)$	$ul(A)$	$ul(B)$	-	$ul(D)$

- Relaxed plan extraction

g	Open	Closed	RPlan
-	{p(D)}	{}	{}
p(D)	{t(D), p(T)}	{p(D)}	{ul(D)}
t(D)	{p(T), t(C)}	{p(D), t(D)}	{dr(C, D), ul(D)}
t(C)	{p(T), t(B)}	{p(D), t(D), t(C)}	{dr(B, C), dr(C, D), ul(D)}
t(B)	{p(T)}	{p(D), t(D), t(C), t(B)}	{dr(A, B), dr(B, C), dr(C, D), ul(D)}
p(T)	{}	{p(D), t(D), t(C), t(B), p(T)}	{lo(C), dr(A, B), dr(B, C), dr(C, D), ul(D)}

- Relaxed plan heuristics $h^{FF} = 1 + 1 + 1 + 1 + 1 = 5$

Width-based Planning

Iterated Width IW

- Uses the idea of novelty w : $w(s)$ is the size of the smallest subset of atoms in s that is true for the first time in the search
- $IW(k)$ is a breadth-first search that prunes generated states s with $w(s) > k$
- IW is an iterative/sequence of calls to $IW(k)$ for $k = 0, 1, 2, \dots$ over problem P until problem solved or i exceeds number of variables in problem

Properties of IW

- The order of search node expansion can affect which nodes are pruned
- $IW(k)$ expands at most $O(n^k)$ states

Example

Left->right order

- $IW(1)$: {1,2,3}
- $IW(2)$: {1,2,3,4}
- $IW(3)$: {1,2,3,4}

Right->left order

- $IW(1)$: {1,3}
- $IW(2)$: {1,3,2,5}
- $IW(3)$: {1,3,2,5,4}

