COMP90054 – Week 6 tutorial

Last updated: 3 April 2023

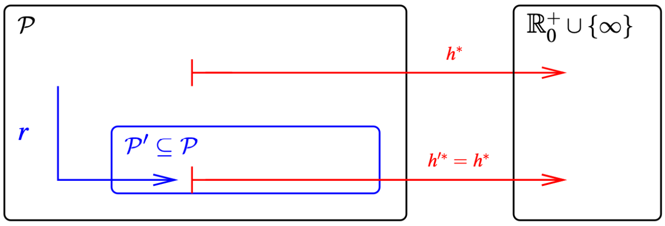
Delete relaxation

Objective

- Find an estimate of the perfect heuristic h^* for STRIPS problem $\mathcal{P}$

Delete relaxation

- Idea: Removing all delete effects of the operators
    - "What was once true remains true forever"
    - State dominance: $s' \supseteq s \rightarrow s'$ dominates s
- Super script "+" → delete relaxed



where, for all $\Pi \in \mathcal{P}$, $h^*(r(\Pi)) \leq h^*(\Pi)$.

For $h^+ = h^* \circ r$**:**

- Problem $\mathcal{P}$: All STRIPS planning tasks.
- Simpler problem $\mathcal{P}'$: All STRIPS planning tasks with empty deletes.
- Perfect heuristic h'^* for $\mathcal{P}'$: Optimal plan cost $= h^*$ on $\mathcal{P}'$.
- Transformation r: Drop the deletes.

→ Is this a native relaxation? Yes.
→ Is this relaxation efficiently constructible? Yes.
→ Is this relaxation efficiently computable? No.

- h^+ is the optimal delete relaxation heuristics (cost of an optimal relaxed plan)
- Delete relaxation is admissible, so can use h^+ to estimate h^*
- Optimal delete relaxed planning is NP-complete, so it is hard to find h^+

Delete and precondition relaxation

- Relaxed problem does not care about preconditions and deletes
- $h^{\text{pre\&del}}$, as the optimal pre&del relaxation heuristics, is admissible, so can use $h^{\text{pre\&del}}$ to estimate h^*
- Optimal pre&del relaxed planning is NP-complete (subset sum problem), so it is hard to find $h^{\text{pre\&del}}$
- Can use $h^{\text{goal count}}$ to estimate $h^{\text{pre\&del}}$, but it is neither admissible nor consistent
    - Action cost < 1 → inadmissible
    - One cost action achieving two goals → inconsistent

Estimate h^+

- $h(s, g)$ estimates the cost of making all propositions $p \in g$ true at state s
  - The original representation $h(s)$ is essentially $h(s, G)$

Definition (h^{add}). *Let $\Pi = (F, A, c, I, G)$ be a STRIPS planning task. The additive heuristic h^{add} for Π is the function $h^{\text{add}}(s) := h^{\text{add}}(s, G)$ where $h^{\text{add}}(s, g)$ is the point-wise greatest function that satisfies $h^{\text{add}}(s, g) =$*

$$\begin{cases} 0 & g \subseteq s \\ min_{a \in A, g \in add_a} c(a) + h^{\text{add}}(s, pre_a) & |g| = 1 \\ \sum_{g' \in g} h^{\text{add}}(s, \{g'\}) & |g| > 1 \end{cases}$$

Definition (h^{max}). *Let $\Pi = (F, A, c, I, G)$ be a STRIPS planning task. The max heuristic h^{max} for Π is the function $h^{\text{max}}(s) := h^{\text{max}}(s, G)$ where $h^{\text{max}}(s, g)$ is the point-wise greatest function that satisfies $h^{\text{max}}(s, g) =$*

$$\begin{cases} 0 & g \subseteq s \\ min_{a \in A, g \in add_a} c(a) + h^{\text{max}}(s, pre_a) & |g| = 1 \\ \max_{g' \in g} h^{\text{max}}(s, \{g'\}) & |g| > 1 \end{cases}$$

- Two ways to estimate h^+: h^{add} and h^{max}
  - h^{add} sums up the cost of all singleton sub-goals as an estimate
  - h^{max} uses the costliest sub-goal as an estimate
- h^{max} is optimistic: $h^{\text{max}} \leq h^+ \leq h^*$
  - h^{max} is admissible, but the estimate can be too small
- h^{add} is pessimistic: $h^{\text{add}} \geq h^+$, so $\exists \Pi, s$ such that $h^{\text{add}}(s) > h^*(s)$
  - h^{add} yields larger estimate, but no guarantee on admissibility
- Use Bellman-Ford algorithm to calculate h^{add} and h^{max} (the table method)

Bellman-Ford variant computing h^{add} for state s

new table $T_0^{\text{add}}(g)$, for $g \in F$

For all $g \in F$: $T_0^{\text{add}}(g) := \begin{cases} 0 & g \in s \\ \infty & \text{otherwise} \end{cases}$

fn $c_i(g) := \begin{cases} T_i^{\text{add}}(g) & |g| = 1 \\ \sum_{g' \in g} T_i^{\text{add}}(g') & |g| > 1 \end{cases}$

fn $f_i(g) := \min[c_i(g), \min_{a \in A, g \in add_a} c(a) + c_i(pre_a)]$

do forever:

    new table $T_{i+1}^{\text{add}}(g)$, for $g \in F$

    For all $g \in F$: $T_{i+1}^{\text{add}}(g) := f_i(g)$

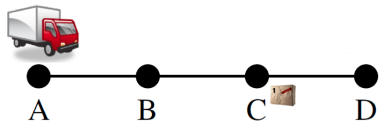
    if $T_{i+1}^{\text{add}} = T_i^{\text{add}}$ **then** stop **endif**

    $i := i + 1$

enddo

- Swap $\Sigma_{g' \in g}$ for $\max_{g' \in g}$ in $c_i(g)$ to compute h^{max}

Logistics example from lecture



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

Content of Tables $T_i^{\mathbf{add}}$: (Table content T_i^1, where different, given in red)

i	$t(A)$	$t(B)$	$t(C)$	$t(D)$	$p(T)$	$p(A)$	$p(B)$	$p(C)$	$p(D)$
0	0	∞	∞	∞	∞	∞	∞	0	∞
1	0	1	∞	∞	∞	∞	∞	0	∞
2	0	1	2	∞	∞	∞	∞	0	∞
3	0	1	2	3	3	∞	∞	0	∞
4	0	1	2	3	3	4	5 (4)	0	7 (4)
5	0	1	2	3	3	4	5 (4)	0	7 (4)

h^{add}

[0] Initial state: t(A), p(C)
[1] t(B): dr(A,B) + sum(T_0(t(A))) -> 1 + 0 = 1
[2] t(C): dr(B,C) + sum(T_1(t(B))) -> 1 + 1 = 2
[3] t(D) : dr(C,D) + sum(T_2(t(C))) -> 1 + (2 + 0) = 3
p(T) : load + sum(T_2(t(C)) + T_2(p(C))) -> 1 + (2 + 0) = 3
[4] p(A) : unload + sum(T_3(t(A)) + T_3(p(T))) -> 1 + (0 + 3) = 4
p(B) : unload + sum(T_3(t(B)) + T_3(p(T))) -> 1 + (1 + 3) = 5
p(D) : unload + sum(T_3(t(D)) + T_3(p(T))) -> 1 + (3 + 3) = 7

h^{max}

[0] Initial state: t(A), p(C)
[1] t(B): dr(A,B) + max(T_0(t(A))) -> 1 + 0 = 1
[2] t(C): dr(B,C) + max(T_1(t(B))) -> 1 + 1 = 2
[3] t(D) : dr(C,D) + max(T_2(t(C))) -> 1 + max(2 , 0) = 3
p(T) : load + max(T_2(t(C)) + T_2(p(C))) -> 1 + max(2 , 0) = 3
[4] p(A) : unload + max(T_3(t(A)) + T_3(p(T))) -> 1 + max(0 , 3) = 4
p(B) : unload + max(T_3(t(B)) + T_3(p(T))) -> 1 + max(1 , 3) = 4
p(D) : unload + max(T_3(t(D)) , T_3(p(T))) -> 1 + max(3 , 3) = 4