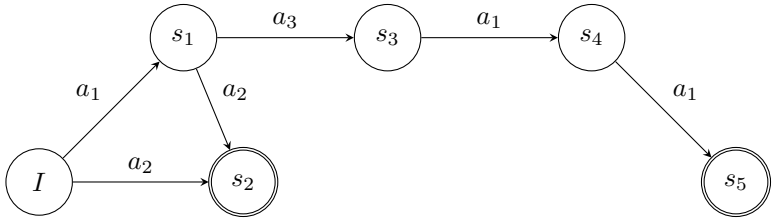
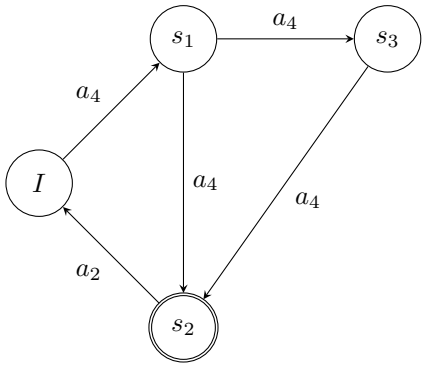
Exercise 1

We have a planning problem with four actions, a_1, a_2, a_3, a_4, and have computed several abstract transition systems as follows (self-loop transitions are ommited):

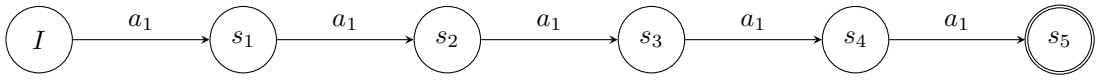
- Θ_{α_1}



- Θ_{α_2}



- Θ_{α_3}



i) Which abstractions are orthogonal?

ii) What is the best heuristic value that we can obtain for the initial state using the notion of orthogonal abstractions?

iii) Consider the abstractions in the given order, compute a saturated cost partitioning. For each abstraction, mark what is the cost function, as well as the heuristic value for each abstract state. What value do we obtain in total for the initial state?

iv) Consider the abstractions in the reverse order (from bottom to top), compute a saturated cost partitioning. For each abstraction, mark what is the cost function, as well as the heuristic value for each abstract state. What value do we obtain in total for the initial state? Is it the same as in the other order?

v) Repeat the saturated cost partitioning with order from top to bottom, but now, instead of assuming that all actions cost 1, our original problem had the following costs: $c(a_1) = 10$, $c(a_2) = 2$, $c(a_3) = 1$, $c(a_4) = 5$.

vi) Is the obtained heuristic with the method of orthogonallity and with the method of saturated cost partitioning admissible? Justify your answer.

Solution:

i) α_2 and α_3 are orthogonal

ii) $\max(1, 2 + 5) = 7$

iii) We specify the costs used by each heuristic, ommiting those where the value is 0.
costs used on α_1: $c_{\alpha_1}(a_1) = c_{\alpha_1}(a_2) == c_{\alpha_1}(a_3) = 1$
costs used on α_2: $c_{\alpha_2}(a_4) = 1$
$h(I) = 1 + 2 + 0 = 3$

iv) costs used on α_3: $c_{\alpha_1}(a_1) = 1$
costs used on α_2: $c_{\alpha_2}(a_4) = 1$
costs used on α_1: $c_{\alpha_1}(a_2) = c_{\alpha_1}(a_3) = 1$
$h(I) = 1 + 2 + 5 = 8$

v) costs used on α_1: $c_{\alpha_1}(a_1) = 10, c_{\alpha_1}(a_2) = 2, c_{\alpha_1}(a_3) = 0$
costs used on α_2: $c_{\alpha_2}(a_4) = 5$

vi) Yes, both methods guarantee that the cost of each action is only counted once, so that we can add the values and the estimate remains admissible.

Exercise 2

Consider a Blocksworld task with three blocks A, B, C with initial state and goal as shown in Figure 1.

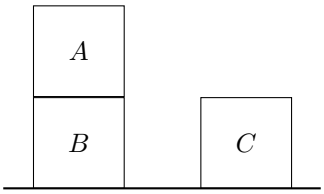


Figure 1: Initial state of the running example.

The encoding has three types of variables, $on(x)$ to describe which block is on top of block x (so on "points upwards"); $ontable(x)$, to say if a block is currently on the table, and $holding$, describing which block is currently held in the hand. The $\perp$ symbol represents the fact that *nothing* is on top of a block, respectively *nothing* is held in the hand.

There are four types of actions, $(un)stack(x, y)$, to $(un)stack$ block x onto/away from block y; $pickup(x)$ and $drop(x)$ to $pickup$ block x from the table into the hand, or $drop$ it to the table if it is held in the hand. In the initial state depicted in Figure 1, the only two applicable actions are $unstack(A, B)$ and $pickup(C)$.

Formally, the FDR encoding of the task, $\Pi = (V, A, I, G)$, is defined as follows:

- Variables: $V = \{on(x), holding, ontable(x)\}$ for $x \in \{A, B, C\}$ with domains
  - $D(ontable(x)) = \{\top, \perp\}$,
  - $D(on(x)) = \{A, B, C, \perp\} \setminus \{x\}$, and
  - $D(holding) = \{A, B, C, \perp\}$.
- Actions: $A = \{stack(x, y), unstack(x, y), pickup(x), drop(x)\}$ (uniform action costs)
  - $stack(x, y)$, with $x \neq y \in \{A, B, C\}$
    pre: $\{holding = x, on(y) = \perp\}$,
    eff: $\{holding = \perp, on(y) = x\}$

  - $unstack(x, y)$, with $x \neq y \in \{A, B, C\}$
    pre: $\{holding = \bot, on(y) = x, on(x) = \bot\}$,
    eff: $\{holding = x, on(y) = \bot\}$
  - $pickup(x)$, with $x \in \{A, B, C\}$
    pre: $\{holding = \bot, ontable(x) = \top, on(x) = \bot\}$,
    eff: $\{holding = x, ontable(x) = \bot\}$
  - $drop(x)$, with $x \in \{A, B, C\}$
    pre: $\{holding = x\}$,
    eff: $\{holding = \bot, ontable(x) = \top\}$

- Initial state:

$$I = \{on(A) = \bot, on(B) = A, on(C) = \bot, holding = \bot,$$
$$ontable(A) = \bot, ontable(B) = \top, ontable(C) = \top\}$$

- Goal: $G = \{on(B) = A, on(C) = B\}$

(i) Indicate which of the following pairs of patterns are orthogonal. For those that are not orthogonal, explain why in a short sentence:

a) $\{\{holding, on(A)\}, \{ontable(C)\}\}$.
b) $\{\{on(B)\}, \{on(C)\}\}$.
c) $\{\{holding, on(A)\}, \{holding, ontable(C)\}\}$.
d) $\{\{ontable(A), on(B)\}, \{ontable(B), on(A)\}\}$.

(ii) For one of the pairs of patterns that are not orthogonal, is there a cost-partitioning that will make them additive? If yes, indicate what cost partitioning you would use to obtain a heuristic that is as good as possible, and explain why you choose it. Only consider 0-1 cost-partitionings where actions can only have a cost of 0 or 1, so you do not need to consider assigning fractional costs.

Solution:

(i) a) $\{\{holding, on(A)\}, \{ontable(C)\}\}$: No, anything with holding is not orthogoanl to anything else
b) $\{\{on(B)\}, \{on(C)\}\}$. Yes, actions put/remove a block only in one place
c) $\{\{holding, on(A)\}, \{holding, ontable(C)\}\}$: No, they both have holding
d) $\{\{ontable(A), on(B)\}, \{ontable(B), on(A)\}\}$: Yes, actions put/remove a block only in one place

(ii)
- $\{\{holding, on(A)\}, \{ontable(C)\}\}$: pattern 2: operators with cost 1: drop(C). All other operators are irrelevant for this pattern so their cost can be assigned to pattern 1.
- $\{\{holding, on(A)\}, \{holding, ontable(C)\}\}$: pattern 2: operator drop(C) (possibly pick(C), unstack(C, X) too), which is needed to achieve the goal. pattern 1: everything else.

Exercise 3

- Goal: A and B both true.
- Initial state: A and B both false.
- Actions: cA effect A cost 1; cB effect B cost 1; fC effect A and B cost 1.5.
- Patterns $P_1 = \{A\}$ and $P_2 = \{B\}$.

Provide an LP encoding that computes the optimal cost-partitioning. Follow the following steps:

1. Draw the abstract state space of both PDBs.
2. What are the variables of the LP?
3. What is the objective function?
4. What are the constraints?
5. What would be a solution for the linear program?

Solution:

LP variables (ignoring actions not affecting a given PDB)**:** $c_{1,cA}$; $c_{2,cB}$; $c_{1,fC}$, $c_{2,fC}$; h_{P_1}, h_{P_2}; $c_{1,\neg A}$, $c_{1,A}$; $c_{2,\neg B}$, $c_{2,B}$.

LP constraints:

- $c_{1,cA} \leq 1$; $c_{2,cB} \leq 1$; $c_{1,fC} + c_{2,fC} \leq 1.5$
- $h_{P_1} \leq c_{1,A}$; $h_{P_2} \leq c_{2,B}$
- $c_{1,\neg A} = 0$; $c_{1,A} \leq c_{1,\neg A} + c_{1,cA}$; $c_{1,A} \leq c_{1,\neg A} + c_{1,fC}$
  $c_{2,\neg B} = 0$; $c_{2,B} \leq c_{2,\neg B} + c_{2,cB}$; $c_{2,B} \leq c_{2,\neg B} + c_{2,fC}$

Solution maximizing $h_{P_1} + h_{P_2}$**:** For example, $c_{1,cA} = 1$, $c_{1,fC} = 0.75$, $c_{1,A} = 0.75$, $h_{P_1} = 0.75$; $c_{2,cB} = 1$, $c_{2,fC} = 0.75$, $c_{2,B} = 0.75$, $h_{P_2} = 0.75$.