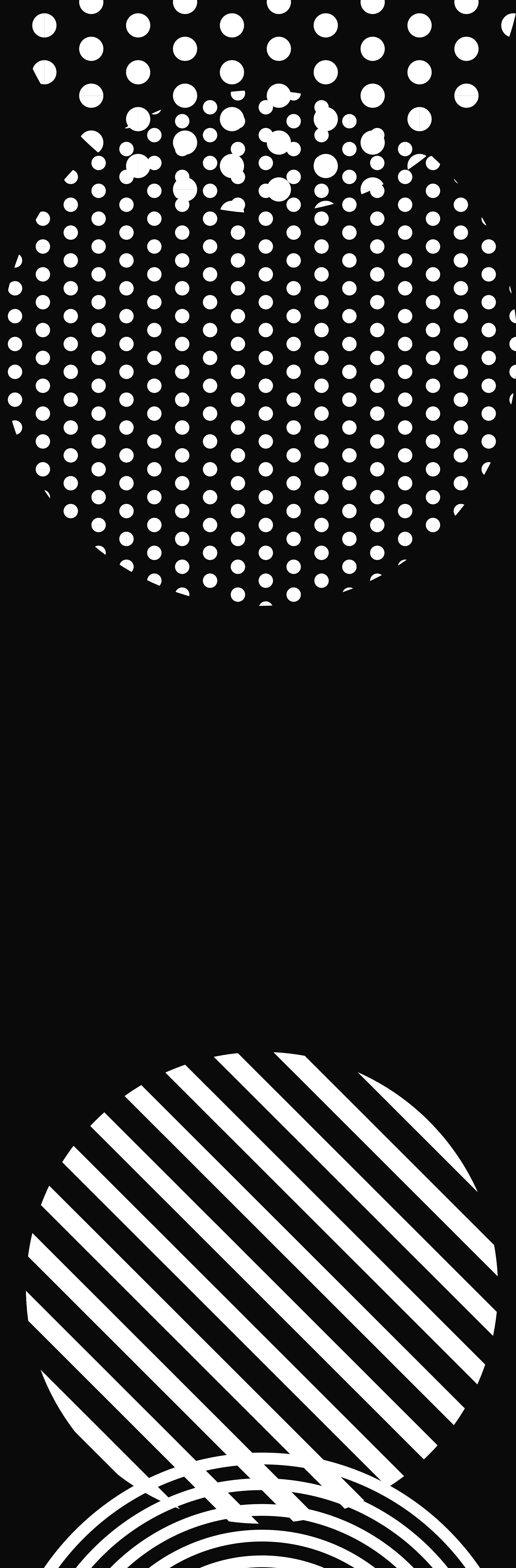


2.1. Let $P_1 = (\Sigma, s_0, g_1)$ and $P_2 = (\Sigma, s_0, g_2)$ be two state-variable planning problems with the same planning domain and initial state. Let $\pi_1 = \langle a_1, \ldots, a_n \rangle$ and $\pi_2 = \langle b_1, \ldots, b_n \rangle$ be solutions for P_1 and P_2, respectively. Let $\pi = \langle a_1, b_1, \ldots, a_n, b_n \rangle$.

(a) If π is applicable in s_0, then is it a solution for P_1? For P_2? Why or why not?

(b) E_1 be the set of all state variables that are targets of the effects in $\text{eff}(a_1), \ldots, \text{eff}(a_n)$, and E_2 be the set of all state variables that are targets of the effects in $\text{eff}(b_1), \ldots, \text{eff}(b_n)$. If $E_1 \cap E_2 = \varnothing$, then is π applicable in s_0? Why or why not?

(c) Let P_1 be the set of all state variables that occur in $\text{pre}(a_1), \ldots, \text{pre}(a_n)$, and P_2 be the set of all state variables that occur in the preconditions of $\text{pre}(b_1), \ldots, \text{pre}(b_n)$. If $P_1 \cap P_2 = \varnothing$ and $E_1 \cap E_2 = \varnothing$, then is π applicable in s_0? Is it a solution for P_1? For P_2? Why or why not?

$\boxed{a}$ $P_1 = (\Sigma, s_0, g_1) \quad P_2 = (\Sigma, s_0, g_2)$

$\Pi_1 = \langle a_1, a_2, \ldots, a_n \rangle \quad \Pi_2 = \langle b_1, b_2, \ldots, b_n \rangle$

$\Pi = \langle a_1, b_1, a_2, b_2, \ldots, a_n, b_n \rangle$

Π_1 APP $S_0 \Rightarrow \exists\, S_1, S_2, \ldots, S_n$ TC $\gamma(S_{i-1}, a_i) = S_i$

Π_2 APP $S_0 \Rightarrow \exists\, S_1', S_2', \ldots, S_n'$ TC $\gamma(S_{i-1}', b_i) = S_i'$

Π APP $S_0 \Rightarrow \exists\, K_1, K_2, \ldots, K_{2n}$ TC $\gamma(K_{i-1}, m_i) = K_i$

WHERE $m_i = \begin{cases} a_{\frac{i+1}{2}} & i \text{ DISPARI} \\ b_{i/2} & i \text{ PARI} \end{cases}$

HENCE

$\gamma(S_0, a_1) = K_1 = S_1$

$\gamma(\underset{S_1}{\underset{\parallel}{K_1}}, b_1) =$ OTHER THINGS

SAME DOING WITH AN EXAMPLE: THE ACTION I SHOULD PERFORM IN A CERTAIN STATE MAY NOT BE FEASIBLE BECAUSE OF CHANGES IN STATE.

$\boxed{b}$ NO, IT IS NOT BECAUSE EVENTHOUGH THE PLANS DO NOT INTERFERE IN EFF VARIABLES, THEY MAY CHANGE PRE VAR OF THE OTHER PLAN

$\boxed{c}$ SAME AS BEFORE (I THINK...)

I THINK THAT THE ONLY WAY IS

$$P_1 \cap E_2 = \phi$$
$$P_2 \cap E_1 = \phi$$
$$E_1 \cap E_2 = \phi$$

2.2. Let S be the state-variable state space discussed in Example 2.7. Give a set of restrictions such that s is a state of S if and only if it satisfies those restrictions.

2.15. Let P be a planning problem in which the action templates and initial state are as shown in Figure 2.16, and the goal is $g = \{\mathsf{loc(c1)} = \mathsf{loc2}\}$. In the Run-Lazy-Lookahead algorithm, suppose the call to Lookahead(P) returns the following solution plan:

$$\pi = \{\mathsf{take(r1,loc1,c1)}, \mathsf{move(r1,loc1,loc2)}, \mathsf{put(r1,loc2,c1)}\}.$$

(a) Suppose that after the actor has performed take(r1,loc1,c1) and move(r1,loc1,loc2), monitoring reveals that c1 fell off of the robot and is still back at loc1. Tell what will happen, step by step. Assume that Lookahead(P) will always return the best solution for P.

(b) Repeat part (a) using Run-Lookahead.

(c) Suppose that after the actor has performed take(r1,loc1,c1), monitoring reveals that r1's wheels have stopped working, hence r1 cannot move from loc1. What should the actor do to recover? How would you modify Run-Lazy-Lookahead, Run-Lookahead, and Run-Concurrent-Lookahead to accomplish this?

EXAMEN ?

SI EJECUTAR (a) FALLA, PARA DETECTARLO PUEDO:
- COMPARAR ESTADO ANTES Y DESPUES
- EJECUTAR RETURNS ALGO
- ESTADO INDETERMINADO

$g = \{ CAR(R_1) = C_1 \; ; \; LOC(R_1) = d_3 \}$

- $LOAD(R_1, C_1, d_1)$ 1, 2, $\Rightarrow$ NO
- $LOAD(R_1, C_1, d_2)$ 1, 2, $\Rightarrow$ NO
- $PUT(R_2, C_1, d_3)$ $\Rightarrow$ NO
- $MOVE(R_1, d_1, d_3)$ 1, 2, 3 $\Rightarrow$ SÍ
- $MOVE(R_1, d_3, d_1)$ $\Rightarrow$ NO
- $MOVE(R_1, d_2, d_3)$ 1, 2, 3 $\Rightarrow$ SÍ

move(r,l,m)
  prec: loc(r)=l, adjacent(l,m)
  efec: loc(r) $\leftarrow m$

load(r,c,l)
  prec: cargo(r)=nil, loc(r)=l, loc(c)=l
  efec: cargo(r) $\leftarrow c$, loc(c) $\leftarrow r$

put(r,l,c)
  prec: loc(r)=l, loc(c)=r
  efec: cargo(r) $\leftarrow$ nil, loc(c) $\leftarrow l$

Ejemplo:

- $g = \{$loc(c1)=r1$\}$
- ¿Qué es γ^{-1}(g, load(r1,c1,d3))?
- ¿Qué es γ^{-1}(g, load(r2,c1,d1))?

$LOAD(R_1, C_1, d_3)$ 1, 2, 3 $\Rightarrow$ RELEVANT !!

$\gamma^{-1}(g, LOAD(R_1, C_1, d_3))$

$= \{ CARGO(R_1) = NIL, \; LOC(R_1) = d_3, \; LOC(C_1) = d_3 \}$

move(r,l,m)
  prec: loc(r)=l, adjacent(l,m)
  efec: loc(r) $\leftarrow m$

load(r,c,l)
  prec: cargo(r)=nil, loc(r)=l,
loc(c)=l
  efec: cargo(r) $\leftarrow c$, loc(c) $\leftarrow r$

put(r,l,c)
  prec: loc(r)=l, loc(c)=r
  efec: cargo(r) $\leftarrow$ nil, loc(c) $\leftarrow l$

2.7. Figure 2.16 shows a planning problem involving two robots whose actions are controlled by a single actor.

(c) Compute the values of $h^{\text{add}}(s_0)$ and $h^{\max}(s_0)$.

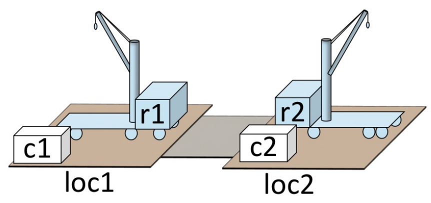
(e) Compute the value of $h^{\text{FF}}(s_0)$.

take(r, l, c)
  pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
  eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
  pre: loc$(r) = l$, pos$(c) = r$
  eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
  pre: loc$(r) = l$
  eff: loc$(r) \leftarrow m$

(a) action templates



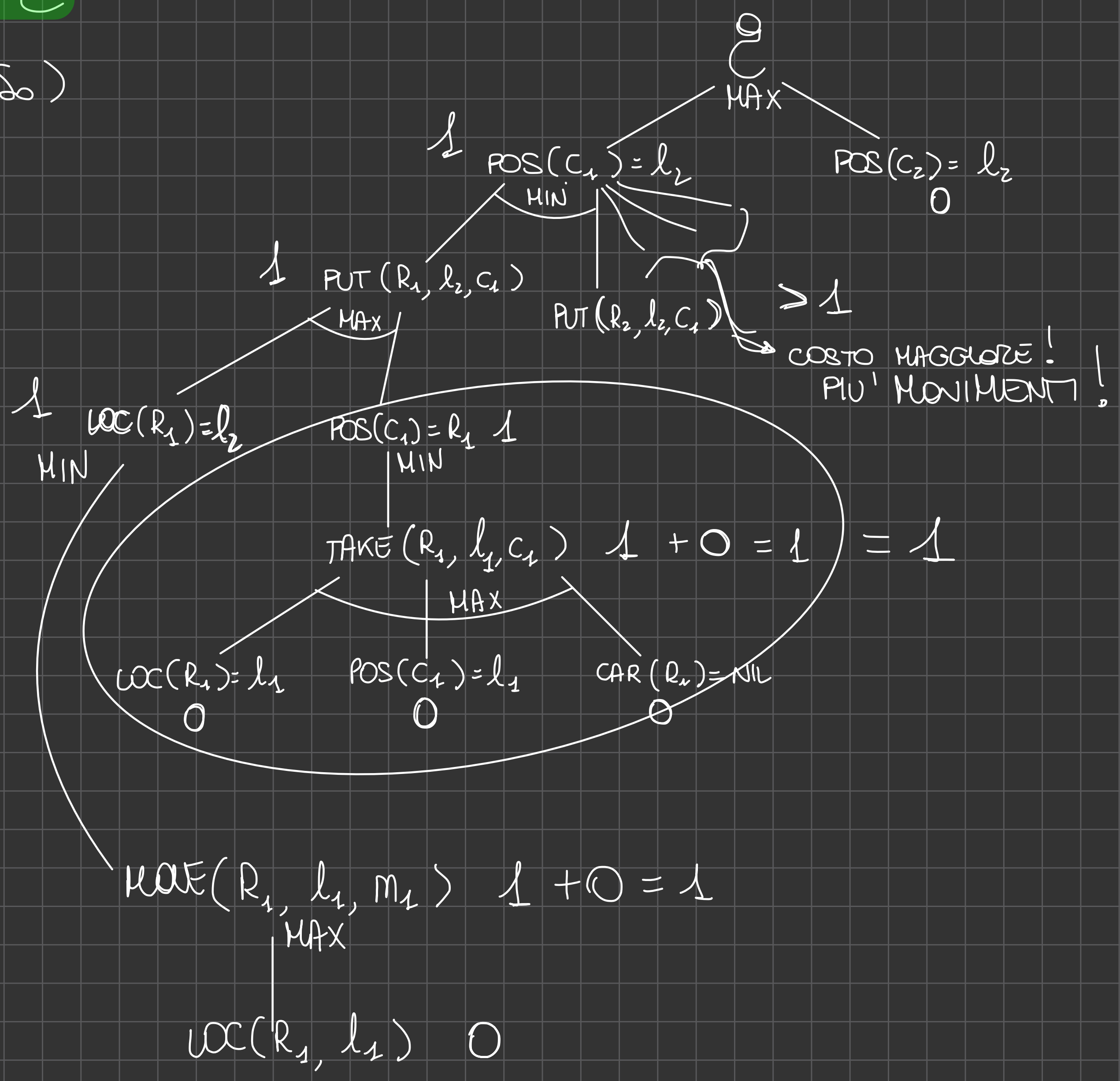
$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

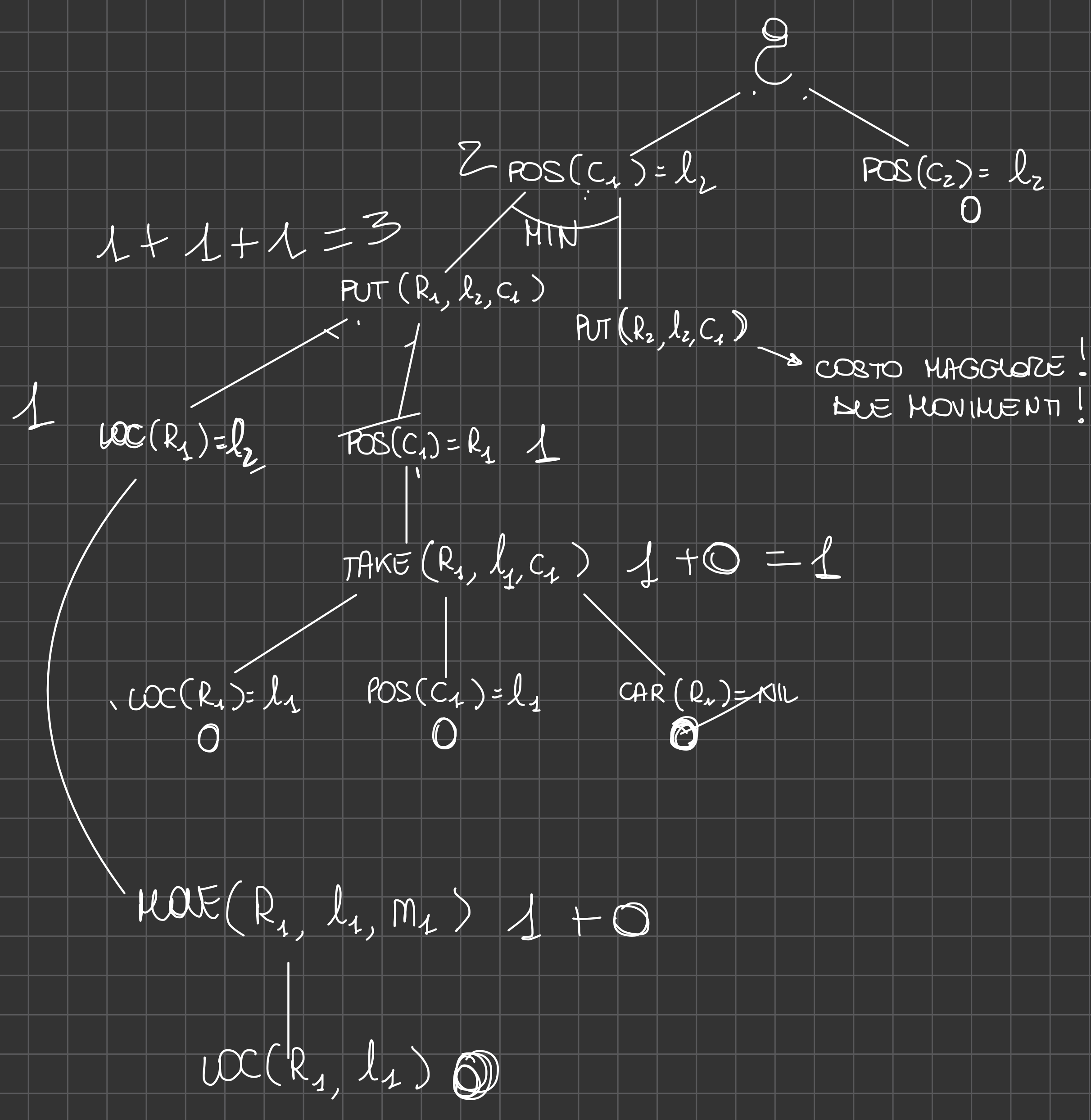
PUNTO C

$h^{MAX}(S_0)$



$\Rightarrow h^{MAX}(S_0) = 1$

$h^{ADD}(S_0) = 3$



PUNTO e

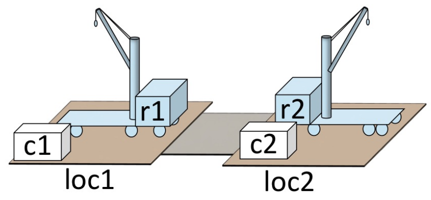
$h^{FF}(s_0)$

take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$,
cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

$L(R_1) = l_1$
$L(R_2) = l_2$
$C(R_1) = N$
$C(R_2) = N$
$P(C_1) = l_1$
$P(C_2) = l_2$

$\xrightarrow{TAKE(R_1, l_1, C_1)}$ $C(R_1) = C_1$, $P(C_1) = R_1$

$\xrightarrow{TAKE(R_2, l_2, C_2)}$ $C(R_2) = C_2$, $P(C_2) = R_2$

$\xrightarrow{MOVE(R_1, l_1, l_2)}$ $L(R_1) = l_2$

$\xrightarrow{MOVE(R_2, l_2, l_1)}$ $L(R_2) = l_1$

$\longrightarrow$ $L(R_1) = l_1$, $L(R_2) = l_2$, $C(R_1) = N$, $C(R_2) = N$, $P(C_1) = l_1$, $P(C_2) = l_2$

$\xrightarrow{PUT(R_1, l_2, C_1)}$ $POS(C_1) = l_2$

$C(R_1) = C_1$
$P(C_1) = R_1$
$C(R_2) = C_2$
$P(C_2) = R_2$
$L(R_1) = l_2$
$L(R_2) = l_1$
$L(R_1) = l_1$
$L(R_2) = l_2$
$C(R_1) = N$
$C(R_2) = N$
$P(C_1) = l_1$
$P(C_2) = l_2$

$h^{FF}(s_0) = 3$

2.8. Here is a state-variable version of the problem of swapping the values of two variables. The set of objects is B = *Variables* $\cup$ *Numbers*, where *Variables* = {foo, bar, baz}, and *Numbers* = $\{0, 1, 2, 3, 4, 5\}$. There is one action template:

$$\begin{aligned}&\mathsf{assign}(x_1, x_2, n)\\ &\quad \text{pre: } \mathsf{value}(x_2) = n\\ &\quad \text{eff: } \mathsf{value}(x_1) \leftarrow n\end{aligned}$$

- *Node selection.* Select a node $(\pi, s) \in$ *Children* that minimizes $h(s)$.
- *Pruning.* Same as in A*.

where Range(x_1) = Range(x_2) = *Variables*, and Range(n) = *Numbers*. The initial state and goal are

$$s_0 = \{\mathsf{value(foo)} = 1, \mathsf{value(bar)} = 5, \mathsf{value(baz)} = 0\};$$
$$g = \{\mathsf{value(foo)} = 5, \mathsf{value(bar)} = 1\}.$$

- *Pruning.* For each node $(\pi, s) \in$ *Children*, if A* has more than one plan that goes to s, then keep only the least costly one. More specifically,

At s_0, suppose GBFS is trying to choose between the actions a_1 = assign(baz,foo,1) and a_2 = assign(foo,bar,5). Let $s_1 = \gamma(s_0, a_1)$ and $s_2 = \gamma(s_0, a_2)$. Compute each pair of heuristic values below, and state whether or not they will produce the best choice.

(a) $h^{\text{add}}(s_1)$ and $h^{\text{add}}(s_2)$.

(b) $h^{\max}(s_1)$ and $h^{\max}(s_2)$.

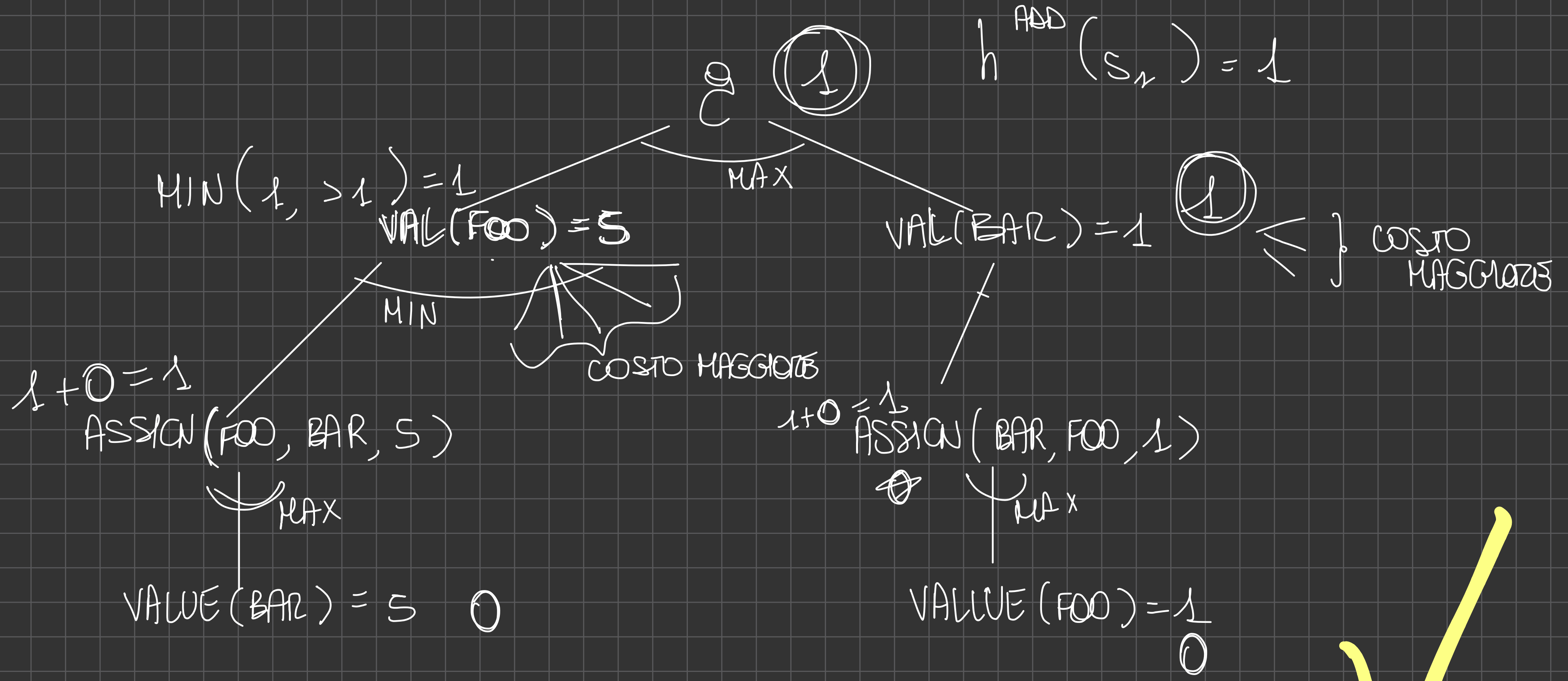
PART b

$s_1 = \gamma(s_0, a_1) = \{ val(foo) = 1, val(baz) = 1, val(bar) = 5 \}$

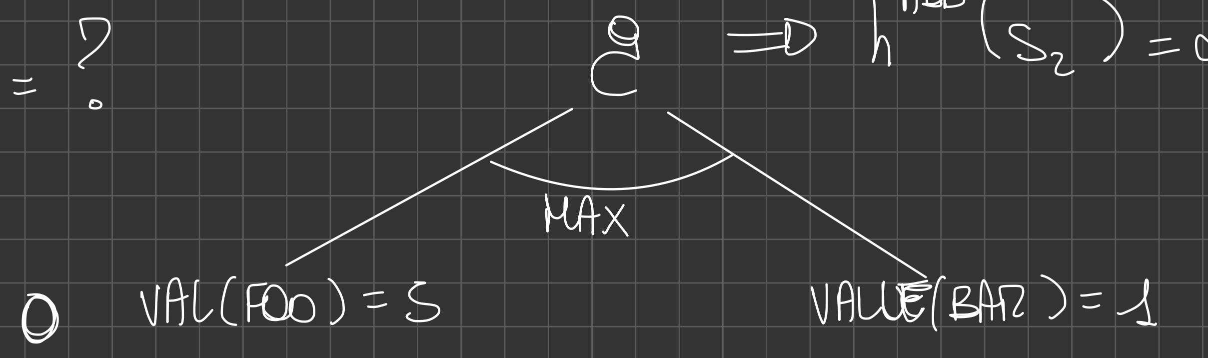
$h^{ADD}(s_1)$

$S_1 = \gamma(S_0, a_1) = \{ VAL(BAZ) = 1;\ VAL(FOO) = 1;\ VAL(BAR) = 5 \}$

$S_2 = \gamma(S_0, a_2) = \{ VAL(FOO) = 5;\ VAL(BAZ) = 0;\ VAL(BAR) = 5 \}$



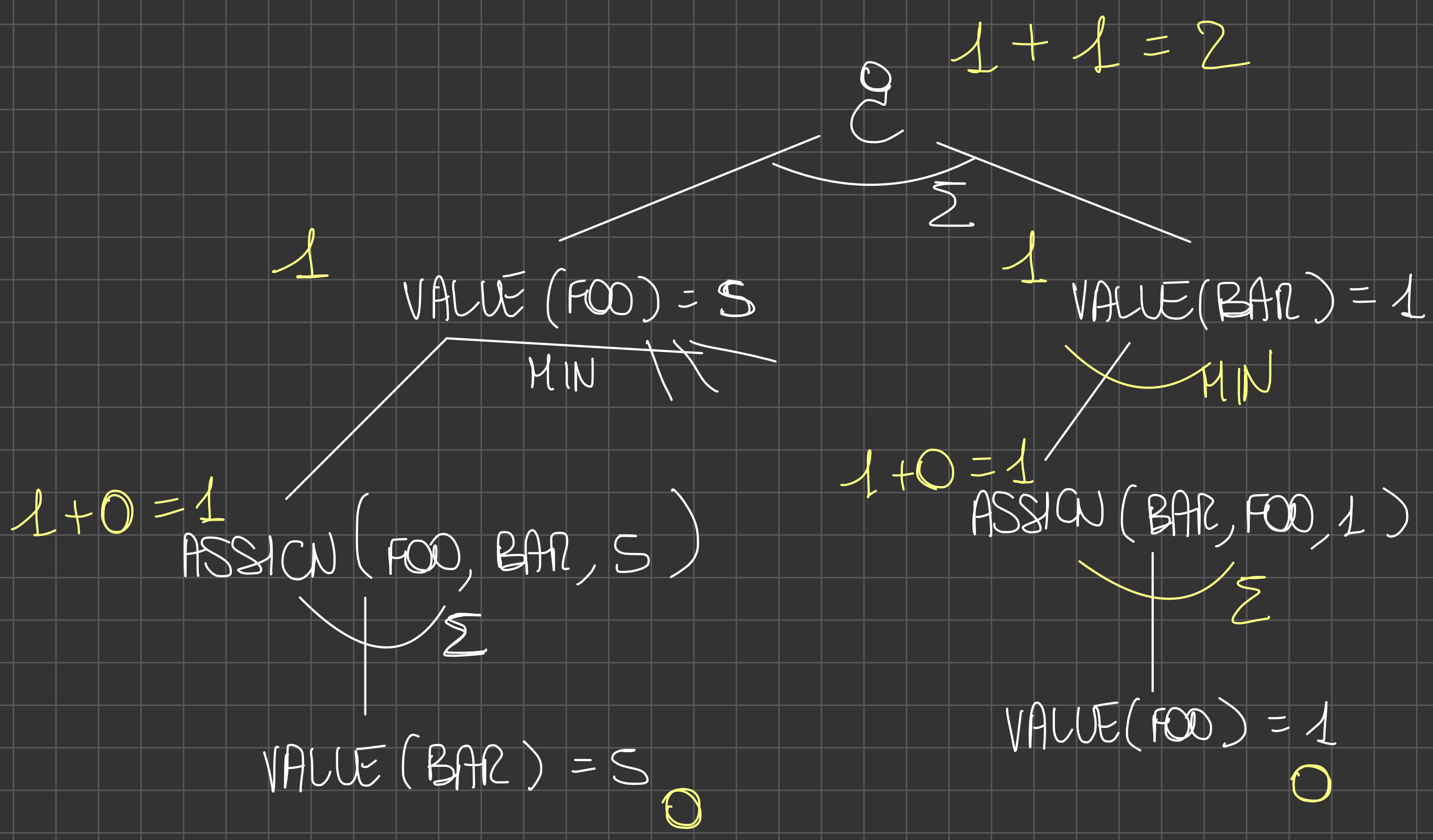
$h^{ADD}(s_2) = ?$ $\Longrightarrow h^{ADD}(s_2) = \infty$



RIVEDERE ∞

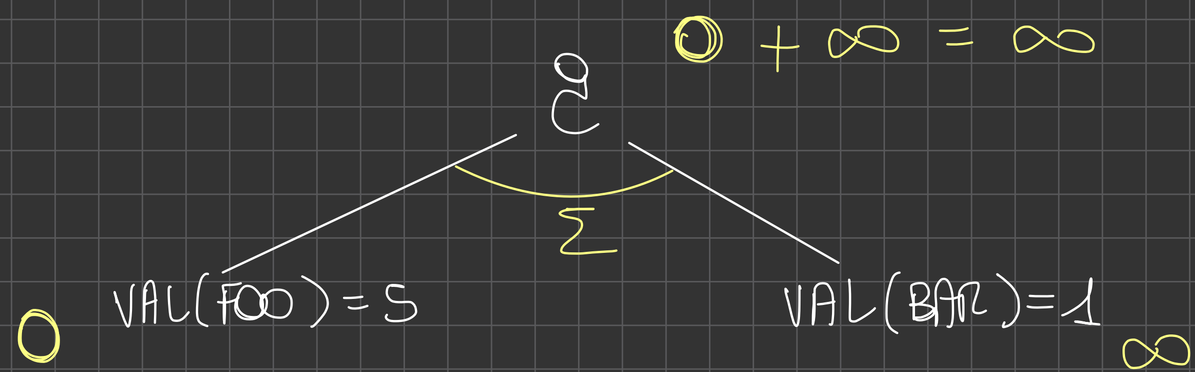
PART a

$h^{ADD}(s_1) = ?$ $\quad s_1 = \gamma(s_0, a_1) = \{ VAL(BAZ) = 1 ; VAL(FOO) = 1 ; VAL(BAR) = 5 \}$



$h^{ADD}(s_2) = ?$ $\quad s_2 = \gamma(s_0, a_2) = \{ VAL(FOO) = 5 ; VAL(BAZ) = 0 ; VAL(BAR) = 5 \}$

$g = \{ VAL(FOO) = 5 ; VAL(BAR) = 1 \}$



IN THIS CASE THEY PRODUCE THE BEST CHOICE BECAUSE S_2 LEADS US TO A STATE FROM WHICH WE CAN NOT REACH g.

✓

2.11. Repeat Exercise 2.8 on the planning problem in Figure 2.18(b), with $s_1 = \gamma(s_0, \mathsf{unstack(c,a)})$ and $s_2 = \gamma(s_0, \mathsf{pickup(b)})$.

(a) $h^{\text{add}}(s_1)$ and $h^{\text{add}}(s_2)$.

(b) $h^{\text{max}}(s_1)$ and $h^{\text{max}}(s_2)$.

pickup(x)
  pre: loc(x) = table, top(x) = nil,
    holding = nil
  eff: loc(x) ← hand, holding ← x

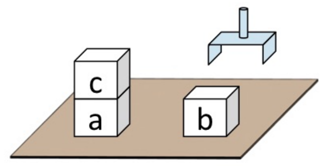
putdown(x)
  pre: holding = x
  eff: loc(x) ← table, holding ← nil

unstack(x, y) **(c, a)**
  pre: loc(x) = y, top(x) = nil,
    holding = nil
  eff: loc(x) ← hand, top(y) ← nil,
    holding ← x

stack(x, y)
  pre: holding = x, top(y) = nil
  eff: loc(x) ← y, top(y) ← x,
    holding ← nil

Range(x) = Range(y) = *Blocks*

(a) action templates



Objects = *Blocks* ∪ {hand, table, nil}
Blocks = {a, b, c}

s_0 = {~~top(a) = c~~, ~~top(b) = nil~~,
  ~~top(c) = nil~~, ~~holding = nil~~,
  loc(a) = table, loc(b) = table,
  ~~loc(c) = a~~}

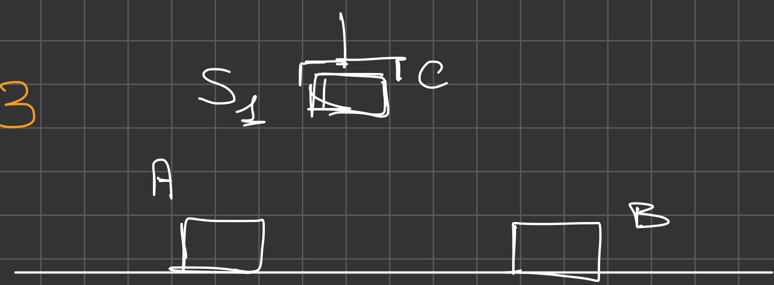
g = {loc(a) = b, loc(b) = c}

(b) objects, initial state, and goal

UNSTACK (C, a)

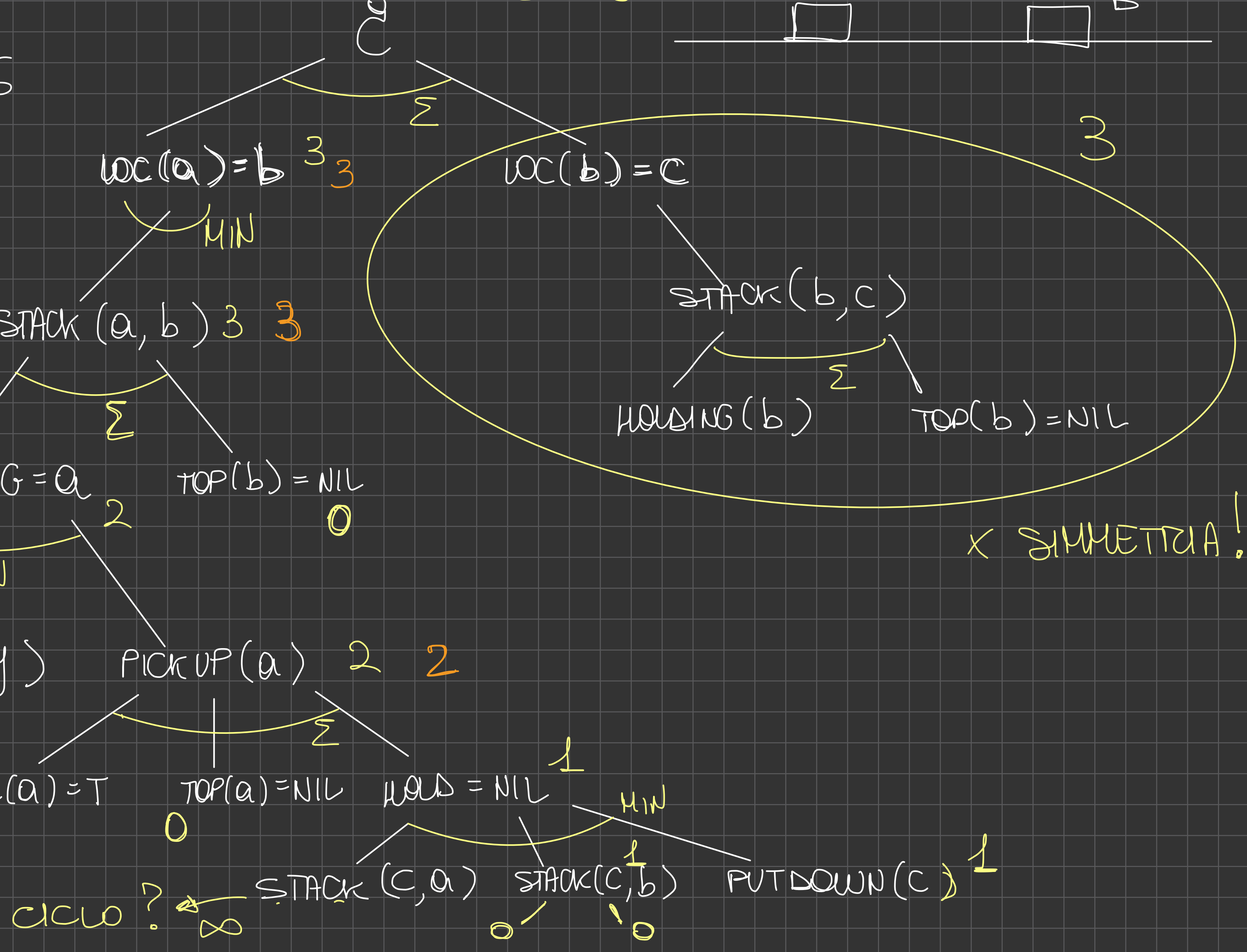
S_1 = { TOP(a) = NIL, TOP(b) = N, TOP(C) = N, HOLD = C, LOC(a) = TAB, LOC(B) = T
LOC(C) = HAND

$S_2 = \gamma(S_0, \text{PICKUP}(b)) =$
{ TOP(a) = C, TOP(b) = N, TOP(C) = N, HOLD = b, LOC(a) = TAB, LOC(B) = HAND,
LOC(C) = a }



PUNTO a-b

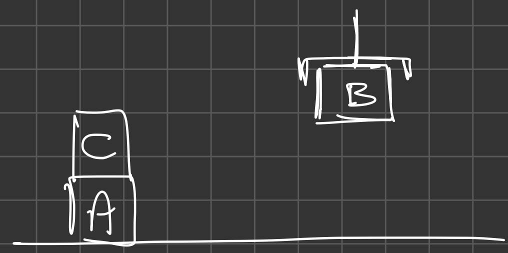
$h^{ADD}(S_1) = 6$



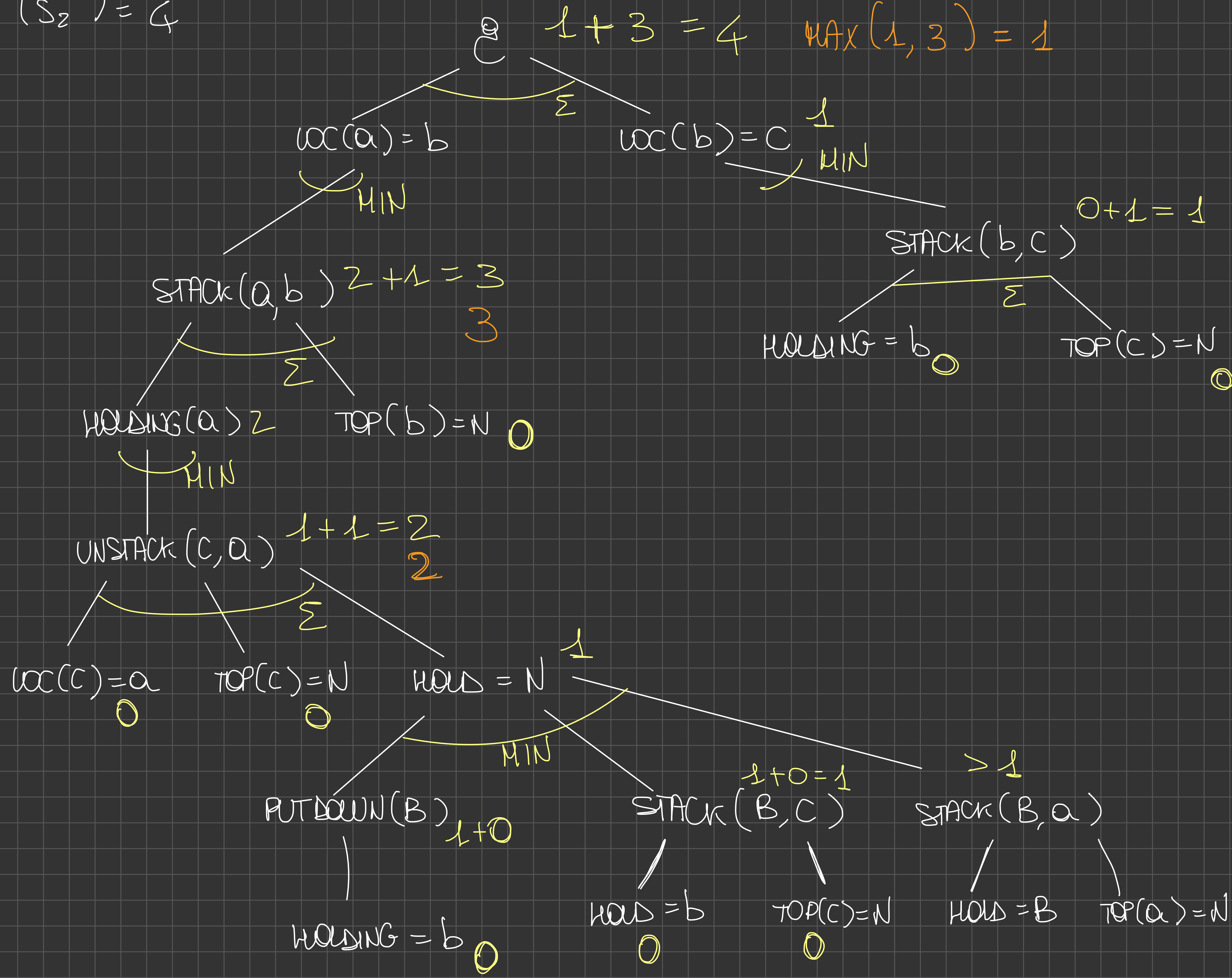
$$S_2 = \gamma(S_0, \text{PICKUP}(b)) =$$

$$\{ TOP(a)=C,\ TOP(b)=N,\ TOP(C)=N,\ HOLD=b,\ LOC(a)=TAB,\ LOC(B)=HAND,$$
$$LOC(C)=a \}$$

$$g = \{ LOC(a)=b;\ LOC(b)=C \}$$



$$h^{ADD}(S_2) = 4$$



2.7. Figure 2.16 shows a planning problem involving two robots whose actions are controlled by a single actor.

(c) Compute the values of $h^{\text{add}}(s_0)$ and $h^{\max}(s_0)$.

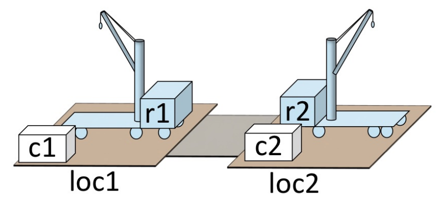
(e) Compute the value of $h^{\text{FF}}(s_0)$.

take(r, l, c)
  pre: loc$(r) = l$, pos$(c) = l$,
    cargo$(r) =$ nil
  eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
  pre: loc$(r) = l$, pos$(c) = r$
  eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
  pre: loc$(r) = l$
  eff: loc$(r) \leftarrow m$

(a) action templates

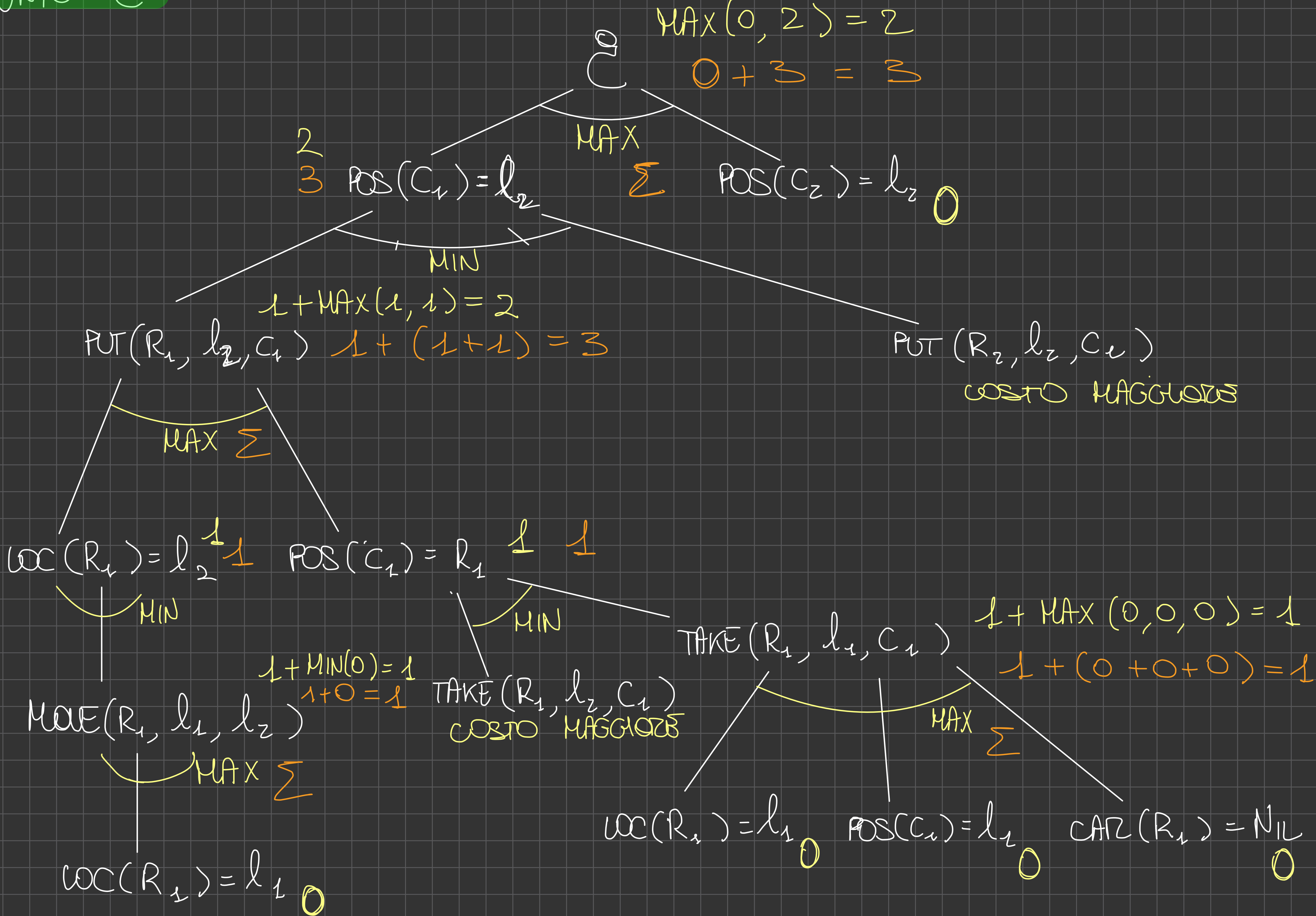


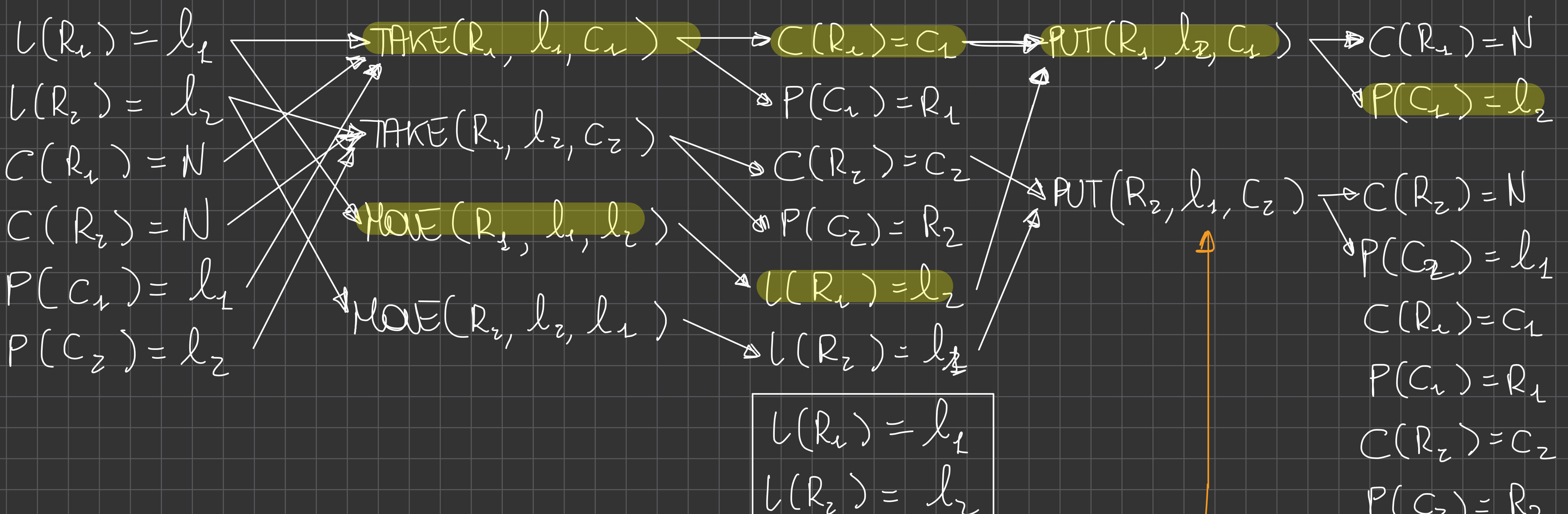
$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2,
cargo(r1) = nil, cargo(r2) = nil,
pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

PUNTO C



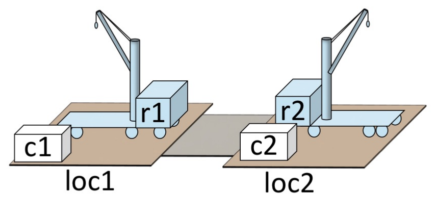


take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

HASTA DESARROJAR TODOS LOS ESTADOS!

SHOULD ALSO EVALUATE
- COME BACK
- TAKE OF OTHER BOX
- PUT IN SAME LOC
- PUT IN OTHER LOC

DO NOT ADD ANYTHING

$h^{FF}(s_0) = 3 \rightarrow$ SUM OF COSTS

POWER *

2.8. Here is a state-variable version of the problem of swapping the values of two variables. The set of objects is $B = \mathit{Variables} \cup \mathit{Numbers}$, where $\mathit{Variables} = \{\mathsf{foo}, \mathsf{bar}, \mathsf{baz}\}$, and $\mathit{Numbers} = \{0, 1, 2, 3, 4, 5\}$. There is one action template:

$$\begin{aligned} &\mathsf{assign}(x_1, x_2, n) \\ &\quad \text{pre: } \mathsf{value}(x_2) = n \\ &\quad \text{eff: } \mathsf{value}(x_1) \leftarrow n \end{aligned}$$

- *Node selection*. Select a node $(\pi, s) \in \mathit{Children}$ that minimizes $h(s)$.
- *Pruning*. Same as in A*.

where $\text{Range}(x_1) = \text{Range}(x_2) = \mathit{Variables}$, and $\text{Range}(n) = \mathit{Numbers}$. The initial state and goal are

- *Pruning*. For each node $(\pi, s) \in \mathit{Children}$, if A* has more than one plan that goes to s, then keep only the least costly one. More specifically,

$$s_0 = \{\mathsf{value}(\mathsf{foo}) = 1, \mathsf{value}(\mathsf{bar}) = 5, \mathsf{value}(\mathsf{baz}) = 0\};$$
$$g = \{\mathsf{value}(\mathsf{foo}) = 5, \mathsf{value}(\mathsf{bar}) = 1\}.$$

At s_0, suppose GBFS is trying to choose between the actions $a_1 = \mathsf{assign}(\mathsf{baz},\mathsf{foo},1)$ and $a_2 = \mathsf{assign}(\mathsf{foo},\mathsf{bar},5)$. Let $s_1 = \gamma(s_0, a_1)$ and $s_2 = \gamma(s_0, a_2)$. Compute each pair of heuristic values below, and state whether or not they will produce the best choice.

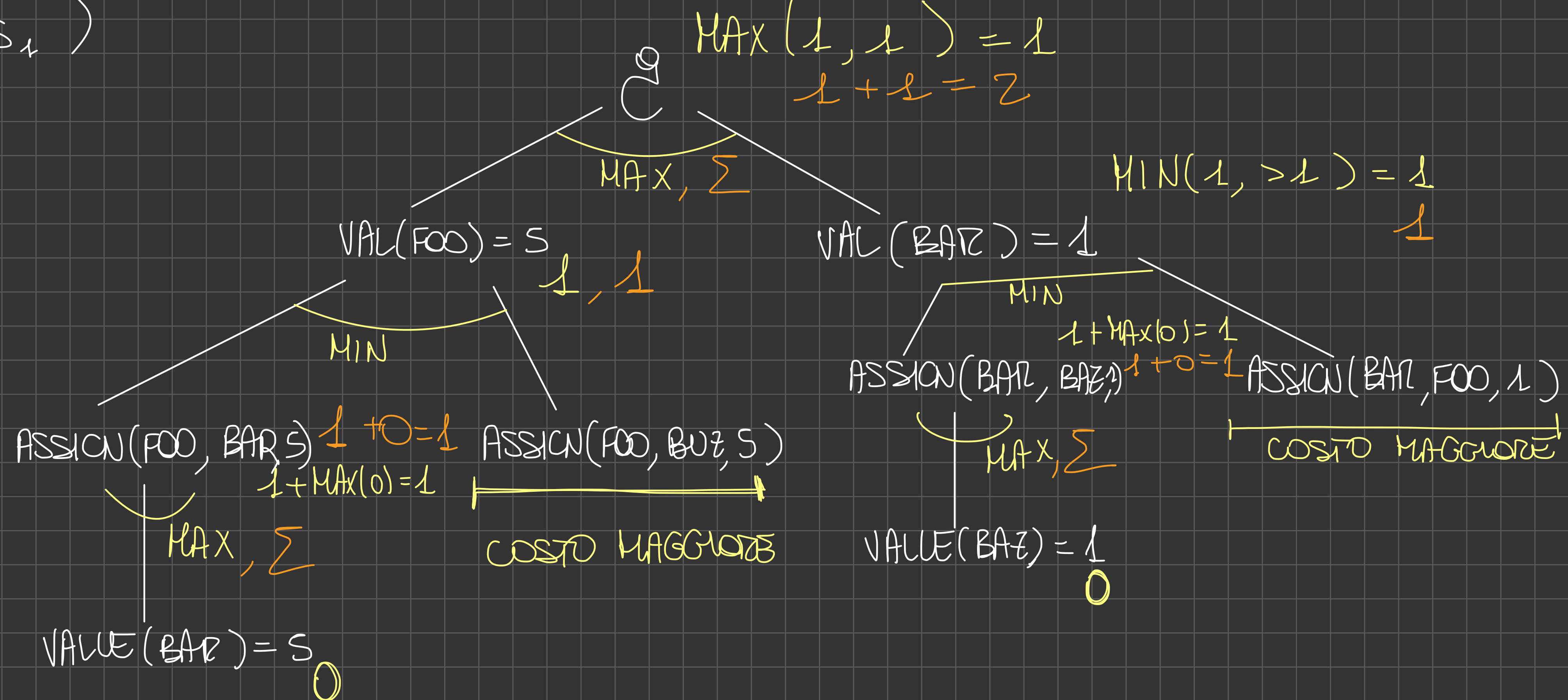
(a) $h^{\text{add}}(s_1)$ and $h^{\text{add}}(s_2)$.

(b) $h^{\text{max}}(s_1)$ and $h^{\text{max}}(s_2)$.

$S_1 = \{$ VAL(FOO) = 1, VAL(BAR) = 5, VAL(BAZ) = 1 $\}$

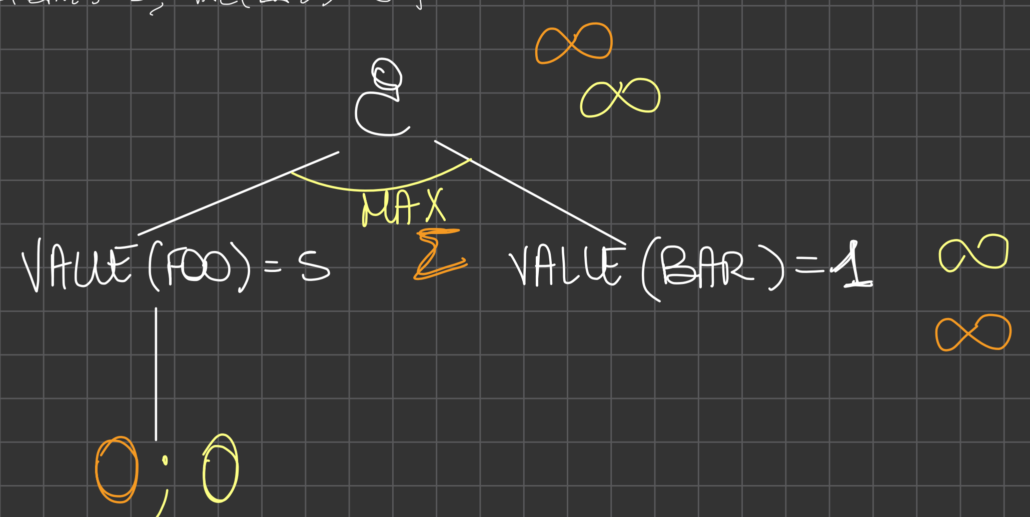
$S_2 = \{$ VAL(FOO) = 5, VAL(BAR) = 5, VAL(BAZ) = 0 $\}$

h(S1)



$S_2 = \{$ VAL(FOO) = 5, VAL(BAR) = 5, VAL(BAZ) = 0 $\}$

h(S2)



$\boxed{h^{FF}}$

$\hat{S}_0$

VAL(FOO) = 1
VAL(BAR) = 5
VAL(BAZ) = 1

$\hat{a}_1$

ASS(FOO, BAR, 5)
ASS(FOO, BAZ, 1)
ASS(FOO, FOO, 1)
ASS(BAR, BAR, 1)
ASS(BAZ, BAZ, 1)
ASS(BAZ, BAR, 5)
ASS(BAZ, FOO, 1)
ASS(BAR, BAZ, 1)
ASS(BAR, FOO, 1)

VAL(FOO, 5)
VAL(BAR, 1)
VAL(BAZ, 5)

$h^{FF}(S_1) = 2$

PARA $h^{FF}(S_2)$ SI HAGO OTRA ACCION LLEGO A OTRO ESTADO IGUAL AL ANTERIOR
$\Rightarrow$ NO SOLUCCION

2.7. Figure 2.16 shows a planning problem involving two robots whose actions are controlled by a single actor.

(c) Compute the values of $h^{\text{add}}(s_0)$ and $h^{\max}(s_0)$.

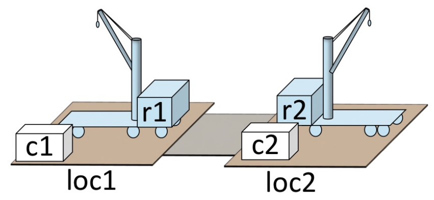
(e) Compute the value of $h^{\text{FF}}(s_0)$.

take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

$\boxed{\text{CON LANDMARK}}$

$\text{COLA} = \{ \text{POS}(C_1) = l_2 ; \underbrace{\cancel{\text{POS}(C_2) = l_2}}_{\in S_0} \}$

$\text{LANDMARK} = \emptyset$

$\text{LANDMARKS} = \{ \text{POS}(C_1) = l_2 \}$

$\text{COLA} = \emptyset$

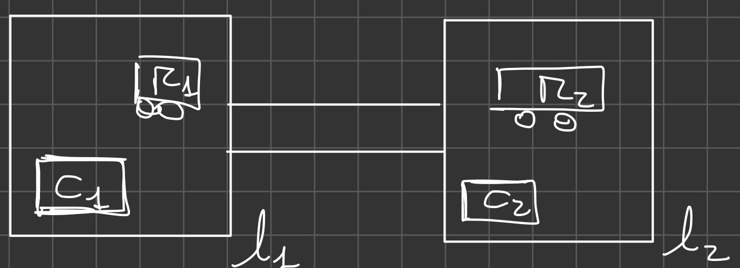
- OBTENER CONJUNTO R PARA ATOMO $\text{POS}(C_1) = l_2$ (TODAS LAS POSIBLES!!)

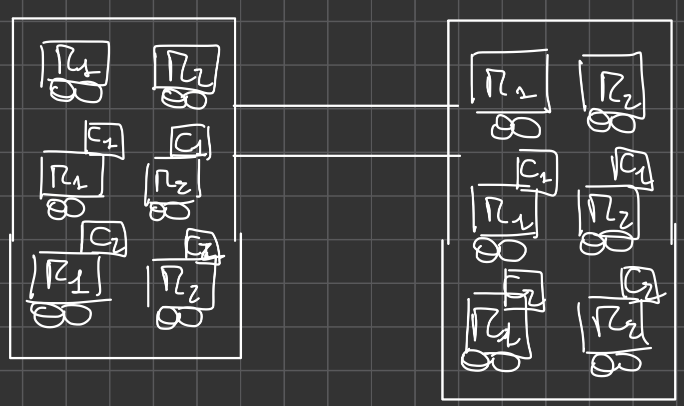
$R = \{ \text{PUT}(R_1, C_1, l_2) ; \text{PUT}(R_2, C_1, l_2) \}$

- OBTENER CONJUNTO N DE ACCIONES r-APLICABLES DESDE $\hat{S}_0$
DOS MANERAS PARA HACERLO

1. VISUALIZAR

$\hat{S}_0$





CUALES DE R SON APLICABLE DESDE $\hat{S}_K$?

LAS DOS DE R $\Rightarrow N = R$

2. RPG

$\hat{S}_0$	$\hat{A}_1$	$\hat{S}_1$
$LOC(R_1) = l_1$	$MOVE(R_1, l_1, l_2)$	$LOC(R_2) = l_1$
$LOC(R_2) = l_2$	$MOVE(R_2, l_2, l_1)$	$LOC(R_1) = l_2$
$CAR(R_1) = N$	$TAKE(R_1, C_1, l_1)$	$CARGO(R_1) = C_1$
$CAR(R_2) = N$	$TAKE(R_2, C_2, l_2)$	$CARGO(R_2) = C_2$
$POS(C_1) = l_1$	(NO PUT!)	$POS(C_1) = R_1$
$POS(C_2) = l_2$		$POS(C_2) = R_2$
		$\boxed{\hat{S}_0}$

$\hat{A}_2$	$\hat{S}_2$	$\hat{A}_3$
$MOVE(R_1, l_2, l_2)$ *	$CARGA(R_1) = C_2$	NADA!
$MOVE(R_2, l_1, l_2)$ *	$CAR(R_2) = C_1$	
$TAKE(R_1, C_2, l_2)$	$POS(C_1) = R_2$	
$TAKE(R_2, C_1, l_1)$	$POS(C_2) = R_1$	
	$\boxed{\hat{S}_1}$	

DESDE $\hat{S}_2$ SON APLICABLES TODAS LAS ACCIONES EN $R \Rightarrow \boxed{R = N}$

- GENERAMOS LANDMARKS

$$PREC[PUT(R_1, C_1, l_2)] = \{ \underbrace{LOC(R_1) = l_2}_{P_1} ; \underbrace{POS(C_1) = R_1}_{P_2} \}$$

$$PREC[PUT(R_2, C_1, l_2)] = \{ \underbrace{\cancel{LOC(R_2) = l_2}}_{q_1} ; \underbrace{POS(C_1) = R_2}_{q_2} \}$$

CUALES SE CUMPLEN EN S_0? $LOC(R_2) = l_2$ SE CUMPLE $\Rightarrow$ QUITAMOS

GENERAMOS TODAS LAS COMBINACIONES

NUEVOS LANDMARK $= \{ P_1 \vee q_2 ; P_2 \vee q_2 \}$

$LOC(R_1) = l_2 \vee POS(C_1) = R_2$

$POS(C_1) = R_1 \vee POS(C_1) = R_2$

• AÑADIMOS A LA COLA

$$COLA = \{(LOC(R_1)=l_2 \lor POS(C_1)=R_2);\ (POS(C_1)=R_1 \lor POS(C_1)=R_2)\}$$

• REPETIMOS

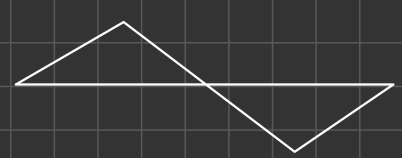
$$LANDMARK = \{(LOC(R_1)=l_2 \lor POS(C_1)=R_2);\ POS(C_1)=l_2\}$$

• IDENTIFICAMOS R

$$R = \{MOVE(R_1, l_1, l_2);\ TAKE(R_2, C_1, l_1);\ \cancel{TAKE(R_2, C_1, l_1)};\ TAKE(R_2, C_1, l_2)\}$$

$\hat{S}_0$	$\hat{A}_1$	$\hat{S}_1$	$\hat{A}_2$
$LOC(R_1)=l_1$	$TAKE(R_1, C_1, l_1)$	$CAR(R_1)=C_1$	$MOVE(R_2, l_2, l_1)$ *
$LOC(C_2)=l_2$	$TAKE(R_1, C_2, l_2)$	$CAR(R_2)=C_2$	$PUT(R_1, C_1, l_1)$ *
$CAR(R_1)=N$	$MOVE(R_2, l_2, l_1)$	$POS(C_1)=R_1$	$PUT(R_2, C_2, l_2)$ *
$CAR(R_2)=N$		$POS(C_2)=R_2$	$PUT(R_2, C_2, l_1)$
$POS(C_1)=l_1$		$LOC(R_2)=l_1$	
$POS(C_2)=l_2$		$\boxed{\hat{S}_0}$	

$\hat{S}_2$	$\hat{A}_3$	$\hat{S}_3$
$POS(C_2)=l_1$	$TAKE(R_1, C_2, l_1)$	$CAR(R_1)=C_2$
$\boxed{\hat{S}_1}$		$POS(C_2)=R_1$
		$\boxed{\hat{S}_2}$



$$N = \{MOVE(R_1, l_1, l_2);\ TAKE(R_2, C_1, l_1)\}$$

$$PREC[MOVE(R_1, l_1, l_2)] = \{LOC(R_1)=l_1\}$$

$$PREC[TAKE(R_2, C_1, l_1)] = \{LOC(R_2)=l_1,\ LOC(C_1)=l_1,\ CAR(R_2)=N\}$$

SE CUMPLE EN S_0 (UNA ES SUFICIENTE)

$\Longrightarrow$ CORTAMOS ALGORITMO Y DEVOLVEMOS LANDMARKS

$$h(s_0) = 2$$

$$LANDMARKS = \{(LOC(R_1)=l_2 \lor POS(C_1)=R_2);\ POS(C_1)=l_2\}$$

DA FARE • 2.11 CON TUTTE LE EURISTICHE

• 2.7 (O 2.8 ?) BUSQUEDA HACIA ATRAS

2.11. Repeat Exercise 2.8 on the planning problem in Figure 2.18(b), with $s_1 = \gamma(s_0, \text{unstack(c,a)})$ and $s_2 = \gamma(s_0, \text{pickup(b)})$.

(a) $h^{\text{add}}(s_1)$ and $h^{\text{add}}(s_2)$.

(b) $h^{\text{max}}(s_1)$ and $h^{\text{max}}(s_2)$.

(c)

HACERLO CON LANDMARK TAMBIEN

pickup(x)
  pre: loc(x) = table, top(x) = nil, holding = nil
  eff: loc(x) ← hand, holding ← x

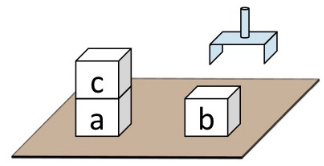
putdown(x)
  pre: holding = x
  eff: loc(x) ← table, holding ← nil

unstack(x, y)
  pre: loc(x) = y, top(x) = nil, holding = nil
  eff: loc(x) ← hand, top(y) ← nil, holding ← x

stack(x, y)
  pre: holding = x, top(y) = nil
  eff: loc(x) ← y, top(y) ← x, holding ← nil

Range(x) = Range(y) = *Blocks*

(a) action templates



Objects = *Blocks* ∪ {hand, table, nil}
Blocks = {a, b, c}

s_0 = {top(a) = c, top(b) = nil, top(c) = nil, holding = nil, loc(a) = table, loc(b) = table, loc(c) = a}

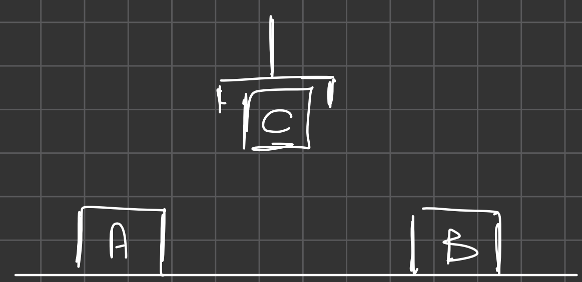
g = {loc(a) = b, loc(b) = c}

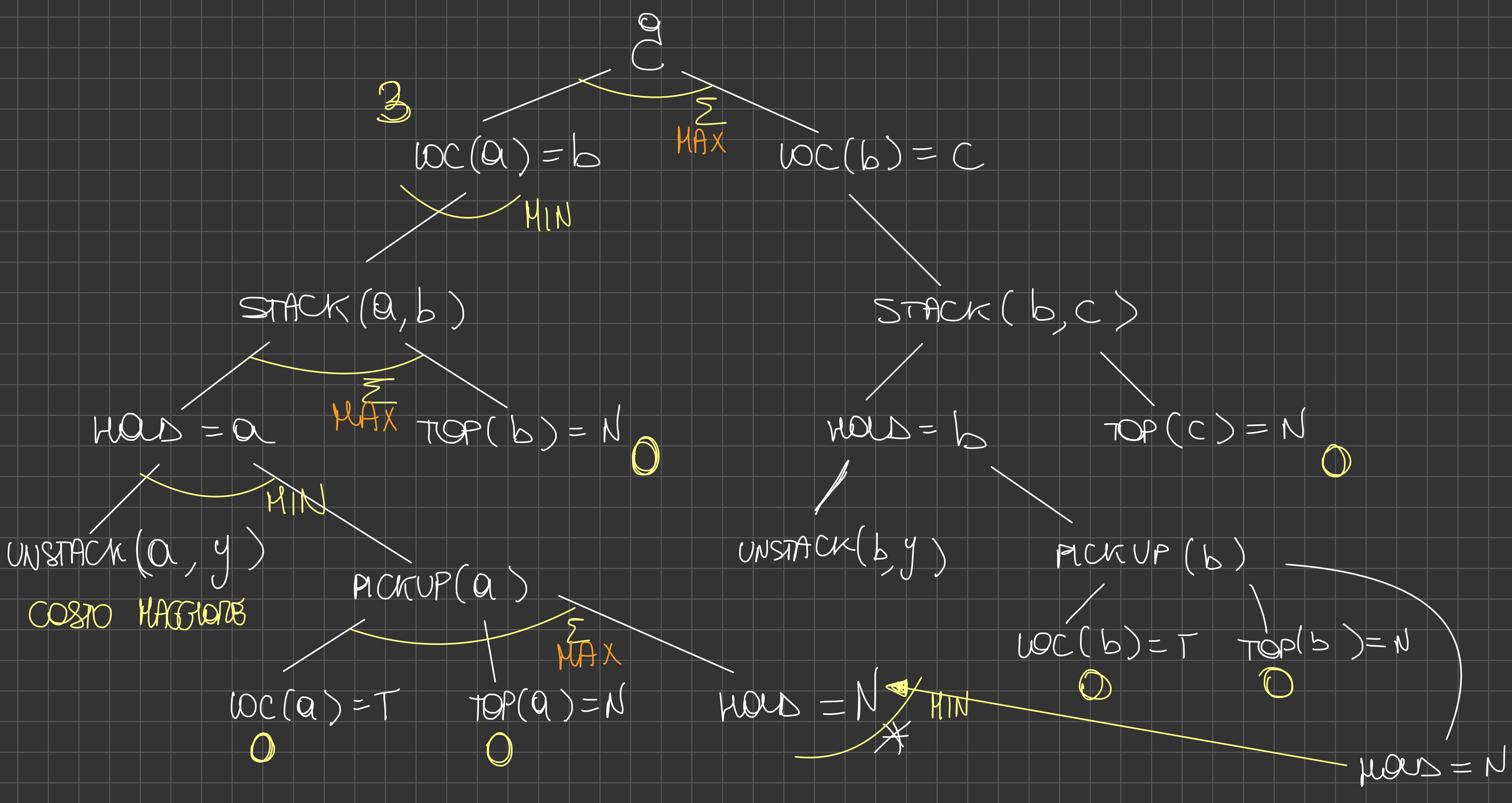
(b) objects, initial state, and goal

$S_1 = \gamma(S_0, \text{UNSTACK}(c, a))$

$S_0 = \{\text{TOP}(a) = C,\ \text{TOP}(b) = N,\ \text{TOP}(C) = N,\ \text{HOLD} = N,\ \text{LOC}(a) = T,\ \text{LOC}(b) = T,\ \text{LOC}(c) = a\}$

$S_1 = \{\text{TOP}(a) = N,\ \text{TOP}(b) = N,\ \text{TOP}(C) = N,\ \text{HOLD} = N,\ \text{LOC}(a) = T,\ \text{LOC}(b) = T,\ \text{LOC}(c) = H\}$





HOLD = N

PUTDOWN(a) ≥ 0 | PUTDOWN(b) ≥ 0 | PUTDOWN(c) | COMBINACIONES DE UNSTACK(x, y) ≥ 0

HOLD = C 0

SI OCURRE UN BUCLE INFINITO LO CORTAMOS Y DAMOS COSTE ∞

h^{ADD} NO ES ADMISIBLE ; ES MEJOR AUNQUE NO SEA ADMISIBLE

h^{MAX} ES ADMISIBLE

LANDMARK

PRIMER RETURN DE LANDMARK ALGORITHM: ES COMO UN "CONTINUE"

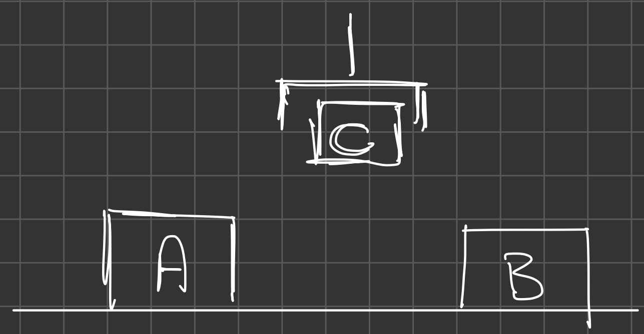
$G = \{ loc(a) = b ; loc(b) = c \}$

$h^{SL}(s_1)$

COLA = $\{ loc(a) = b ; loc(b) = c \}$

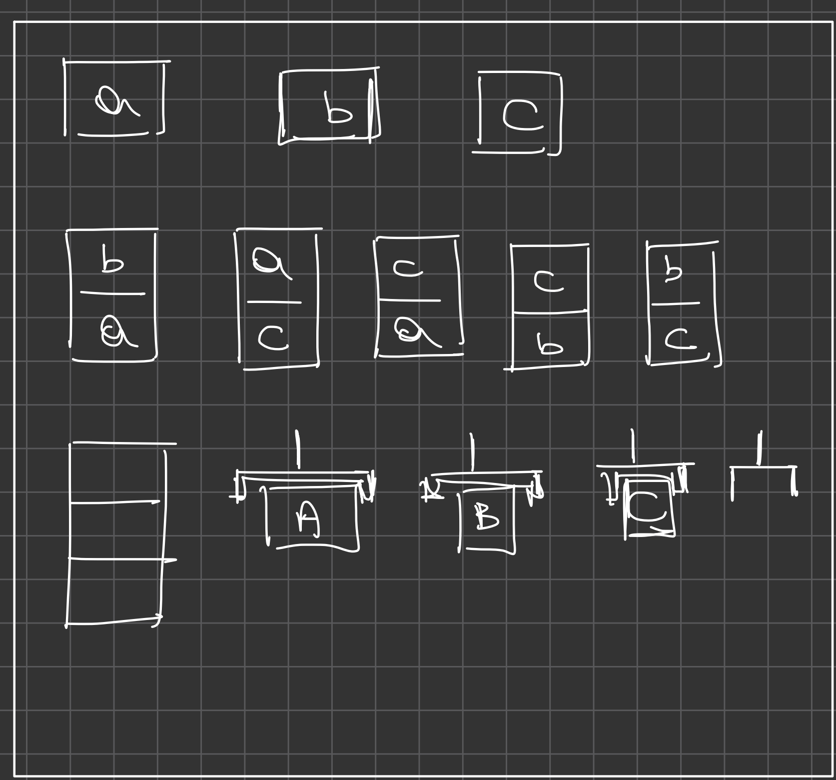
$G_1 = (loc(a) = b)$

LANDMARK = $\{ loc(a) = b \}$



COLA = $\{ loc(b) = c \}$

$R = \{ STACK(a, b) \}$



S_0	A_1	$\hat{S}_1$	A_2
LOC(a) = T	STACK(C, a)	TOP(a) = C	PICKUP(a)
LOC(b) = T	STACK(C, b)	TOP(b) = C	PICKUP(b)
LOC(C) = H	PUTDOWN(C)	LOC(C) = a	PICKUP(C) *
TOP(a) = N		LOC(C) = b	UNSTACK(C, a) *
TOP(C) = N		LOC(C) = T	UNSTACK(C, b) *
$\boxed{\text{TOP(b) = N}}$		HOLD = N	
HOLD = C		$\boxed{S_0}$	

TENGO QUE VERIFICAR SI LAS PRECONDICIONES PARA R SE CUMPLEN! SI EN UN ESTADO INTERMEDIO SE CUMPLEN PODEMOS CORTAR EL GRAFO

$\hat{S}_2$
LOC(a) = H
LOC(b) = H
$\boxed{\text{HOLD = a}}$
HOLD = b
$\boxed{S_1}$

N = { STACK(a, b) }

PREC (STACK(a, b)) = { ~~TOP(b) = N~~, HOLD = a }

SE CUMPLE EN EL ESTADO QUE ESTAMOS EVALUANDO

COLA = { LOC(b) = C , HOLDING = a }

FINAL 1ª ITERACION

2.8. Here is a state-variable version of the problem of swapping the values of two variables. The set of objects is $B = \mathit{Variables} \cup \mathit{Numbers}$, where $\mathit{Variables} = \{\mathsf{foo}, \mathsf{bar}, \mathsf{baz}\}$, and $\mathit{Numbers} = \{0, 1, 2, 3, 4, 5\}$. There is one action template:

$$\begin{aligned}&\mathsf{assign}(x_1, x_2, n)\\ &\quad \text{pre: } \mathsf{value}(x_2) = n\\ &\quad \text{eff: } \mathsf{value}(x_1) \leftarrow n\end{aligned}$$

where $\text{Range}(x_1) = \text{Range}(x_2) = \mathit{Variables}$, and $\text{Range}(n) = \mathit{Numbers}$. The initial state and goal are

$$s_0 = \{\mathsf{value}(\mathsf{foo}) = 1, \mathsf{value}(\mathsf{bar}) = 5, \mathsf{value}(\mathsf{baz}) = 0\};$$
$$g = \{\mathsf{value}(\mathsf{foo}) = 5, \mathsf{value}(\mathsf{bar}) = 1\}.$$

At s_0, suppose GBFS is trying to choose between the actions $a_1 = \mathsf{assign(baz,foo,1)}$ and $a_2 = \mathsf{assign(foo,bar,5)}$. Let $s_1 = \gamma(s_0, a_1)$ and $s_2 = \gamma(s_0, a_2)$. Compute each pair of heuristic values below, and state whether or not they will produce the best choice.

(a) $h^{\text{add}}(s_1)$ and $h^{\text{add}}(s_2)$.

(b) $h^{\max}(s_1)$ and $h^{\max}(s_2)$.

USING S_1 AND LANDMARKS

$S_0 = \{ \text{VAL(FOO)} = 1, \text{VAL(BAR)} = 5, \text{VAL(BAZ)} = 0 \}$

$S_1 = \{ \text{VAL(FOO)} = 1, \text{VAL(BAR)} = 5, \text{VAL(BAZ)} = 1 \}$

$g = \{ \text{VAL(FOO)} = 5, \text{VAL(BAR)} = 1 \}$

1. $\text{CODA} = \{ \text{VAL(FOO)} = 5, \text{VAL(BAR)} = 1 \}$ $\qquad L = \phi$

$g = \text{VAL(FOO)} \Rightarrow L = \{ \text{VAL(FOO)} = 5 \}$

$R = \{ \underbrace{\text{ASSIGN(FOO, BAR, 5)}}_{\text{SATISFIED IN } S_0 \,!!}, \text{ASSIGN(FOO, BAZ, 5)} \}$

$\Rightarrow$ NEXT

LANDMARKS FINAL

2. $\text{CODA} = \{ \text{VAL(BAR)} = 1 \}$

$g = (\text{VAL(BAR)} = 1) \Rightarrow \boxed{L = \{ \text{VAL(FOO)} = 5; \text{VAL(BAR)} = 1 \}}$

$R = \{ \underbrace{\text{ASSIGN(BAR, BAZ, 1)}}_{\text{SATISFIED IN } S_1}; \text{ASSIGN(BAR, FOO, 1)} \}$

$\Rightarrow$ NEXT

2.7. Figure 2.16 shows a planning problem involving two robots whose actions are controlled by a single actor.

(c) Compute the values of $h^{add}(s_0)$ and $h^{max}(s_0)$.

(e) Compute the value of $h^{FF}(s_0)$.

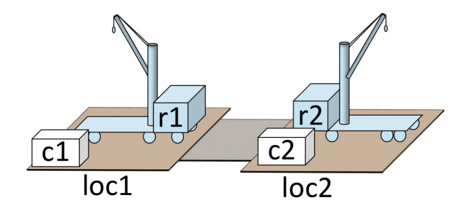
d) COMPUTE $h^{L}(s_0)$

take(r, l, c)
 pre: loc$(r) = l$, pos$(c) = l$,
  cargo$(r) = $ nil
 eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
 pre: loc$(r) = l$, pos$(c) = r$
 eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
 pre: loc$(r) = l$
 eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

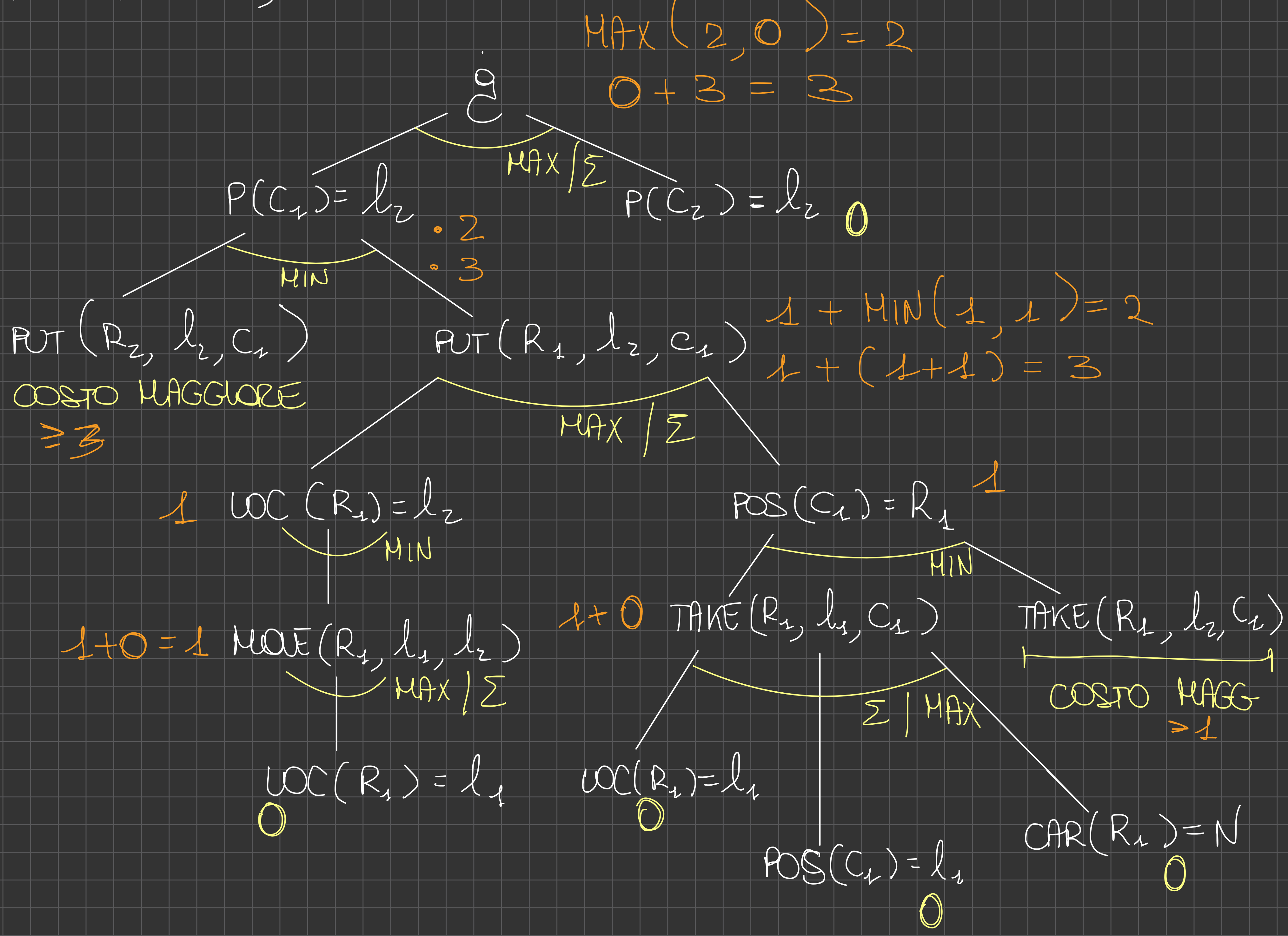
$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

(C)

$S_0 = \{ L(R_1) = L_1,\ L(R_2) = L_2,\ C(R_1) = N,\ C(R_2) = N,$
$POS(C_1) = L_1,\ P(C_2) = L_2 \}$

$g = \{ P(C_1) = L_2,\ P(C_2) = L_2 \}$

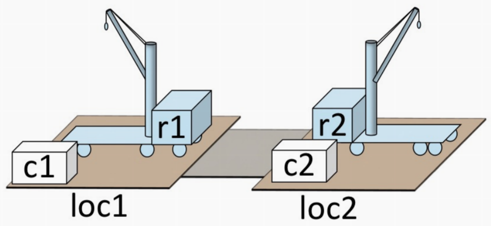


take(r,l,c)
pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r,l,c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r,l,m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

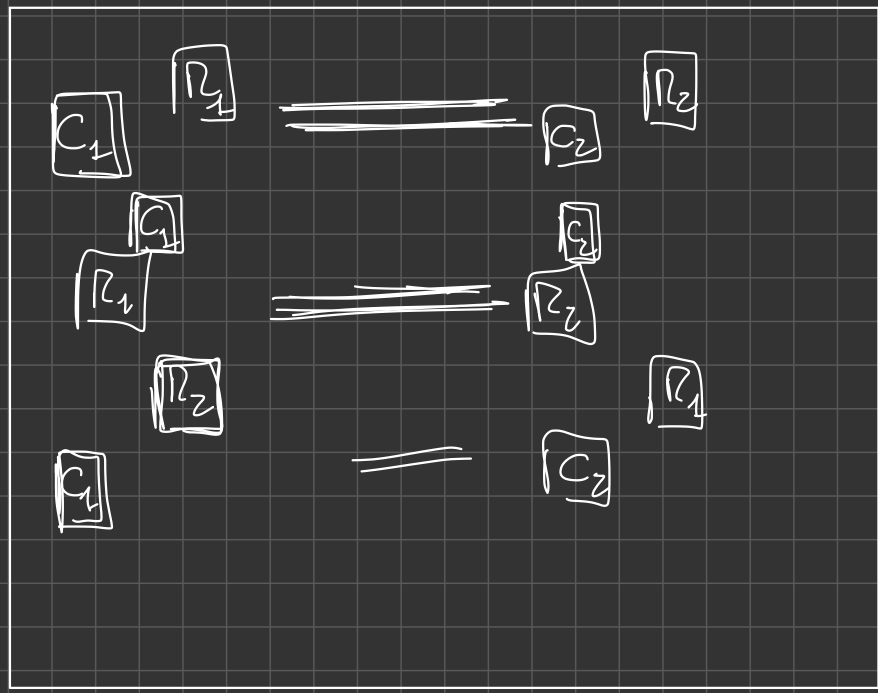
(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal



(e) h^{FF}

$\hat{S}_0$	$\hat{A}_0$	$\hat{S}_1$	$\hat{A}_1$
LOC(R_1) = l_1	TAKE(R_1, l_1, C_1)	CAR(R_1) = C_1	PUT(R_1, l_2, C_1)
LOC(R_2) = l_2	TAKE(R_2, l_2, C_2)	POS(C_1) = R_1	PUT(R_1, l_1, C_1) *
CAR(R_1) = N	MOVE(R_1, l_1, l_2)	CAR(R_2) = C_2	PUT(R_2, l_1, C_2)
CAR(R_2) = N	MOVE(R_2, l_2, l_1)	POS(C_2) = R_2	PUT(R_2, l_2, C_2) *
POS(C_1) = l_1		LOC(R_1) = l_2	MOVE(R_1, l_2, l_1) *
POS(C_2) = l_2		LOC(R_2) = l_1	MOVE(R_2, l_1, l_2) *
		$\boxed{\hat{S}_0}$	TAKE(R_2, l_1, C_1)
			TAKE(R_1, l_2, C_2)

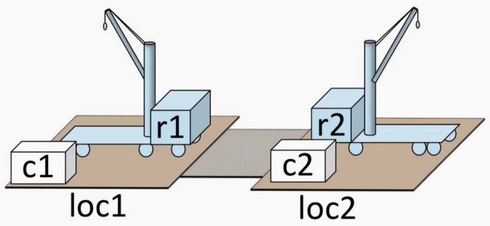
DEVELOP THESE EFFECTS

take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

NO TENEMOS QUE AÑADIRLO POR QUE YA ESTÁ EN S_0

(f) LANDMARK

INIT. COLA $= \{ POS(C_1) = l_2 ; \cancel{POS(C_2) = l_2} \}$

$L = \phi$

1

$g = (POS(C_1) = l_2)$ COLA $= \{ POS(C_2) = l_2 \}$ $L = \{ POS(C_1) = l_2 \}$

$R = \{ PUT(R_1, l_2, C_1) ; PUT(R_2, l_2, C_1) \}$

RELAXED GRAPH

$\hat{S}_0$	$\hat{A}_0$	$\hat{S}_1$	$\hat{A}_1$
$LOC(R_1) = l_1$	$TAKE(R_1, l_1, C_1)$	$CAR(R_1) = C_1$	$MOVE(R_1, l_2, l_1)$ *
$LOC(R_2) = l_2$	$TAKE(R_2, l_2, C_2)$	$CAR(R_2) = C_2$	$MOVE(R_2, l_1, l_2)$ *
$CAR(R_1) = N$	$MOVE(R_1, l_1, l_2)$	$POS(C_1) = R_1$	$TAKE(R_2, l_1, C_1)$
$CAR(R_2) = N$	$MOVE(R_2, l_2, l_1)$	$POS(C_2) = R_2$	$TAKE(R_2, l_2, C_2)$
$POS(C_1) = l_1$		$LOC(R_1) = l_2$	$PUT(R_2, l_1, C_2)$
$POS(C_2) = l_2$		$LOC(R_2) = l_1$	
		$\boxed{\hat{S}_0}$	

$\hat{S}_2$

$CAR(R_2) = C_1$
$CAR(R_1) = C_2$
$POS(C_1) = R_2$
$POS(C_2) = R_1$
$POS(C_2) = l_1$

HERE WE CAN SEE THAT BOTH ACTIONS IN R ARE APLICABLE $\Rightarrow N = R$

$N = \{ PUT(R_1, l_2, c_1) ; PUT(R_2, l_2, c_1) \}$ VERA IN S_0

$$PREC = \{ \underbrace{LOC(R_1) = l_2 ; POS(C_1) = R_1}_{a_1} ; \underbrace{\cancel{LOC(R_2) = l_2} ; POS(C_1) = R_2}_{a_2} \}$$

$$\Phi = \{ \underbrace{LOC(R_1) = l_2 \lor POS(C_1) = R_2}_{p_1} ; \underbrace{POS(C_1) = R_1 \lor POS(C_1) = R_2}_{p_2} \}$$

$$CODA = \{ POS(C_2) = l_2 ; p_1 ; p_2 \}$$

2

$$g = (LOC(R_1) = l_2 \lor POS(C_1) = R_2) \quad L = \{ POS(C_1) = l_1 ; p_1 \}$$

$$R = \{ MOVE(R_1, l_1, l_2) ; TAKE(R_2, l_1, c_1) ; TAKE(R_2, l_2, c_1) \}$$

GRAPH

$\hat{S}_0$	$\hat{A}_0$	$\hat{S}_1$	$\hat{A}_1$
$LOC(R_1) = l_1$	$TAKE(R_1, l_1, C_1)$	$CAR(R_1) = C_1$	$PUT(R_2, l_1, C_2)$
$LOC(R_2) = l_2$	$TAKE(R_2, l_2, C_2)$	$POS(C_1) = R_1$	$PUT(R_1, l_1, C_1)$ *
$CAR(R_1) = N$	$MOVE(R_2, l_2, l_1)$	$CAR(R_2) = C_2$	$PUT(R_2, l_2, C_2)$ *
$CAR(R_2) = N$		$POS(C_2) = R_2$	$MOVE(R_2, l_1, l_2)$ *
$POS(C_1) = l_1$		$LOC(R_2) = l_1$	
$POS(C_2) = l_2$		$\boxed{\hat{S}_0}$	

$\hat{S}_2$	$\hat{A}_2$	$\hat{S}_3$
$POS(C_2) = l_1$	$TAKE(R_1, l_1, C_2)$	$CAR(R_1) = C_2$
$\boxed{\hat{S}_1}$	$TAKE(R_2, l_1, C_2)$ *	$POS(C_2) = R_1$
		$\boxed{\hat{S}_2}$

$$N = \{ MOVE(R_1, l_1, l_2) ; TAKE(R_2, l_1, c_1) ; TAKE(R_2, l_2, c_1) \}$$

2.15. Let P be a planning problem in which the action templates and initial state are as shown in Figure 2.16, and the goal is $g = \{\mathsf{loc(c1)} = \mathsf{loc2}\}$. In the Run-Lazy-Lookahead algorithm, suppose the call to Lookahead(P) returns the following solution plan:

$$\pi = \{\mathsf{take(r1,loc1,c1)}, \mathsf{move(r1,loc1,loc2)}, \mathsf{put(r1,loc2,c1)}\}.$$

(a) Suppose that after the actor has performed take(r1,loc1,c1) and move(r1,loc1,loc2), monitoring reveals that c1 fell off of the robot and is still back at loc1. Tell what will happen, step by step. Assume that Lookahead(P) will always return the best solution for P.

(b) Repeat part (a) using Run-Lookahead.

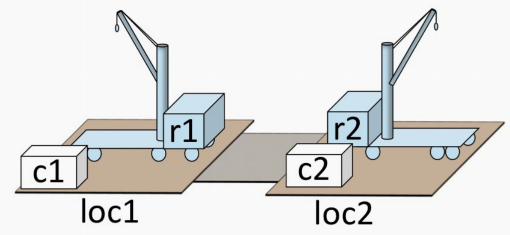
(c) Suppose that after the actor has performed take(r1,loc1,c1), monitoring reveals that r1's wheels have stopped working, hence r1 cannot move from loc1. What should the actor do to recover? How would you modify Run-Lazy-Lookahead, Run-Lookahead, and Run-Concurrent-Lookahead to accomplish this?

take(r, l, c)
 pre: loc(r) $= l$, pos(c) $= l$,
  cargo(r) $=$ nil
 eff: cargo(r) $= c$, pos(c) $\leftarrow r$

put(r, l, c)
 pre: loc(r) $= l$, pos(c) $= r$
 eff: cargo(r) $\leftarrow$ nil, pos(c) $\leftarrow l$

move(r, l, m)
 pre: loc(r) $= l$
 eff: loc(r) $\leftarrow m$

(a) action templates



$s_0 = \{\mathsf{loc(r1)} = \mathsf{loc1}, \mathsf{loc(r2)} = \mathsf{loc2}, \mathsf{cargo(r1)} = \mathsf{nil}, \mathsf{cargo(r2)} = \mathsf{nil}, \mathsf{pos(c1)} = \mathsf{loc1}, \mathsf{pos(c2)} = \mathsf{loc2}\}$

$g = \{\mathsf{pos(c1)} = \mathsf{loc2}, \mathsf{pos(c2)} = \mathsf{loc2}\}$

(b) initial state and goal

(a) $\pi = \langle PUT(R_1, l_2, C_1) \rangle$

WHEN THE FELL IS DETECTED WE HAVE STATE

$S = \{ LOC(R_1) = l_2 ;\ LOC(R_2) = l_2$
$CAR(R_1) = N ;\ CAR(R_2) = N$
$POS(C_1) = l_1 ;\ POS(C_2) = l_2 \}$

WITH LAZY EXECUTION WE HAVE

WHILE $S_0 \notin g$ OR $\pi \neq \langle\rangle$ OR SIMULATE(π, S, Σ) $\neq$ ERR :

 $a \leftarrow$ EXTRACT(π)
 EXECUTE(a)
 S = DETECT STATE

THE SIMULATION ~~W~~ WILL FAIL AND WILL BE PLANNED AN OTHER π LIKE

$\pi' = \langle MOVE(R_1, l_2, l_1);\ TAKE(R_1, l_1, C_1);\ MOVE(R_2, l_2, l_1);$
 $PUT(R_1, l_2, C_2) \rangle$

IF THERE ARE NO OTHER ERRORS THEN THE PLAN WILL BE EXECUTED UNTIL ITS END WITHOUT REPLANNING

(b) USING RUN - LOOKAHEAD

WITH RUN LOOK AHEAD
IT REPLANS π LIKE π'
THEN EXECUTES EACH ACTION IN π ALWAYS REPLANNING

(c)

WE SHOULD ADD:

IF EXECUTION (a) FAILS:
BREAK

THE BREAK LEADS US TO THE PLANNING FUNCTION SO WE WILL HAVE IT REPLANNED AND EVENTUALLY R_2 WILL TAKE THE BOX AFTER R_1 WILL HAVE PUT IT DOWN

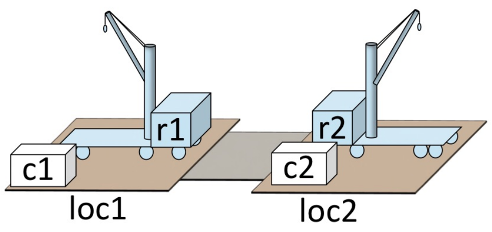
EXERCISE 2.7

take(r, l, c)
  pre: loc$(r) = l$, pos$(c) = l$,
    cargo$(r) =$ nil
  eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
  pre: loc$(r) = l$, pos$(c) = r$
  eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
  pre: loc$(r) = l$
  eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

(b) If we run Backward-search on this problem, how many iterations will the shortest execution traces have, and what plans will they return? For one of them, give the sequence of goals and actions chosen in the execution trace.

3 ITERATIONS   SAME PLAN AS ALWAYS

$g = \{ pos(c_1) = l_2 ; pos(c_2) = l_2 \}$

$A' = \{ put(r_1, l_2, c_1) \quad put(r_2, l_2, c_2) \}$

RELEVANT ACTIONS

$\{ pos(c_2) = l_2 ; loc(r_1) = l_2 ; pos(c_1) = r_1 \}$

$A' = \{ move(r_1, l_1, l_2) ; take(r_1, l_2, c_1) \}$

2.7. Figure 2.16 shows a planning problem involving two robots whose actions are controlled by a single actor.

(c) Compute the values of $h^{\text{add}}(s_0)$ and $h^{\max}(s_0)$.

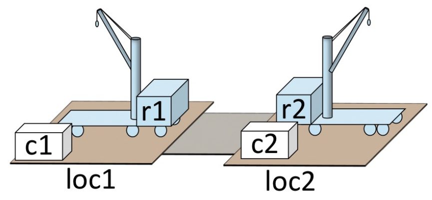
(e) Compute the value of $h^{\text{FF}}(s_0)$.

take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$, cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

d) COMPUTE $h^{L}(s_0)$

(c)

Σ : 3
MAX : 2

g

Σ/MAX

Σ : 3
MAX : 2
POS(C₁) = l₂ — POS(C₂) = l₂ 0

MIN

PUT(R₁, l₂, C₁) — Σ : 1 + 1 = 3, MAX : 1 + 1 = 2 — PUT(R₂, l₁, C₁) HIGHER COST

Σ/MAX

POS(C₁) = R₁ 1 — LOC(R₁) = l₂ 1 — MOVE(R₁, l₁, l₂) 1 — LOC(R₁) = l₁ 0

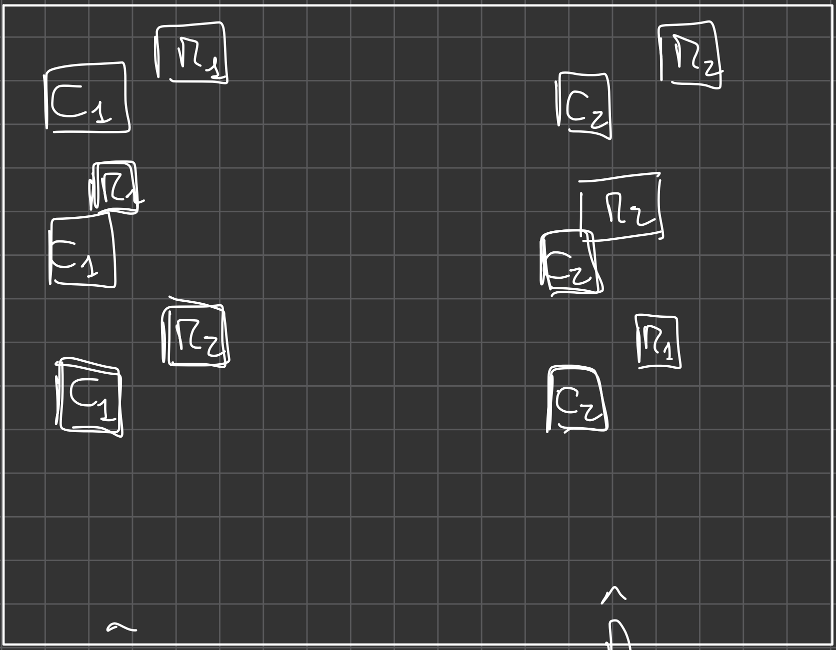
MIN

TAKE(R₁, l₁, C₁) HIGHER COST > 1 — TAKE(R₁, l₂, C₁)

Σ : 1 + 0 = 1
MAX : 1 + 0 = 1

Σ / MAX

LOC(R₁) = l₁ 0 — POS(C₁) = l₁ 0 — CAR(R₁) = N 0

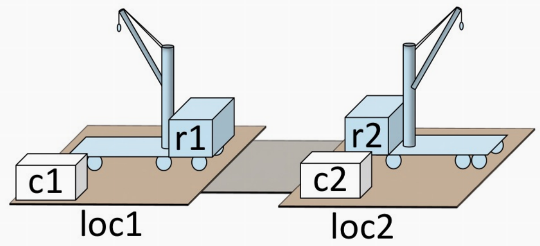


take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$,
cargo$(r) = $ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

$\hat{S}_0$:
- $L(R_1) = l_1$
- $L(R_2) = l_2$
- $C(R_1) = N$
- $C(R_2) = N$
- $P(C_1) = l_1$
- $P(C_2) = l_2$

$\hat{A}_0$:
- TAKE(R_1, l_1, C_1)
- TAKE(R_2, l_2, C_2)
- MOVE(R_1, l_1, l_2)
- MOVE(R_2, l_2, l_1)

$\hat{S}_1$:
- $C(R_1) = C_1$
- $P(C_1) = R_1$
- $C(R_2) = C_2$
- $P(C_2) = R_2$
- $P(R_1) = l_2$
- $P(R_2) = l_1$
- $\boxed{\hat{S}_0}$

$\hat{A}_1$:
- PUT(R_1, l_2, C_1)
- PUT(R_2, l_1, C_2)
- TAKE(R_1, l_2, C_2)
- TAKE(R_2, l_1, C_1)
- MOVE(R_1, l_2, l_1) *
- MOVE(R_2, l_1, l_2) *

$\hat{S}_2$:
- $P(C_2) = l_1$
- $P(C_1) = l_2$
- ⋮
- $\boxed{\hat{S}_1}$

$h^{FF}(s_0) = 3$

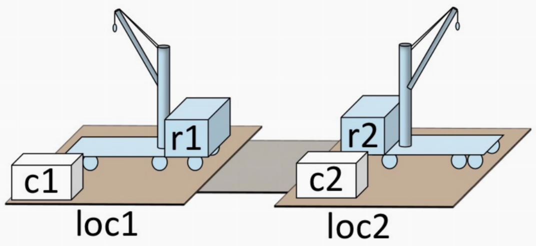
WE CAN STOP WHEN WE FINDS g IN $\hat{S}_i$

take(r, l, c)
pre: loc$(r) = l$, pos$(c) = l$,
cargo$(r) =$ nil
eff: cargo$(r) = c$, pos$(c) \leftarrow r$

put(r, l, c)
pre: loc$(r) = l$, pos$(c) = r$
eff: cargo$(r) \leftarrow$ nil, pos$(c) \leftarrow l$

move(r, l, m)
pre: loc$(r) = l$
eff: loc$(r) \leftarrow m$

(a) action templates



$s_0 = \{$loc(r1) = loc1, loc(r2) = loc2, cargo(r1) = nil, cargo(r2) = nil, pos(c1) = loc1, pos(c2) = loc2$\}$

$g = \{$pos(c1) = loc2, pos(c2) = loc2$\}$

(b) initial state and goal

INIT

GOAL $= \{ POS(C_1) = l_2 \}$ $\quad L = \emptyset$

1. • $g = (POS(C_1) = l_2)$ $\quad L = \{ POS(C_1) = l_2 \}$

• $R = \{ PUT(R_1, l_2, C_1) ; PUT(R_2, l_2, C_1) \}$

• NO ACTION IS SATISFIED IN S_0

• RELAXED GRAPH

$L(R_1) = l_1$
$L(R_2) = l_2$
$C(R_1) = N$
$C(R_2) = N$
$P(C_1) = l_1$
$P(C_2) = l_2$