Artificial Intelligence

2023/2024 Prof: Sara Bernardini

Lab 11: Exam example

Francesco Argenziano
email: argenziano@diag.uniroma1.it

Exercise 1

```
(define (domain safe-party-domain)
	(:requirements :strips)
	(:predicates (at ?x ?y)(ag ?x)(group ?x)(adj ?x ?y)(safe ?x)(movable ?x))

	(:action move
	:parameters (?a ?from ?to)
	:precondition (and (at ?a ?from)(adj ?from ?to)(movable ?to)(ag ?a))
	:effect (and (at ?a ?to)(not(at ?a ?from)))
	)

	(:action divide
	:parameters (?a ?g ?from ?to)
	:precondition (and (group ?g)(at ?g ?from)(ag ?a)(at ?a ?from)(movable ?to)(adj ?from ?to))
	:effect (and (not (movable ?from))(not(movable ?to))(not(at ?g ?from))(safe ?from))
	)
)
```

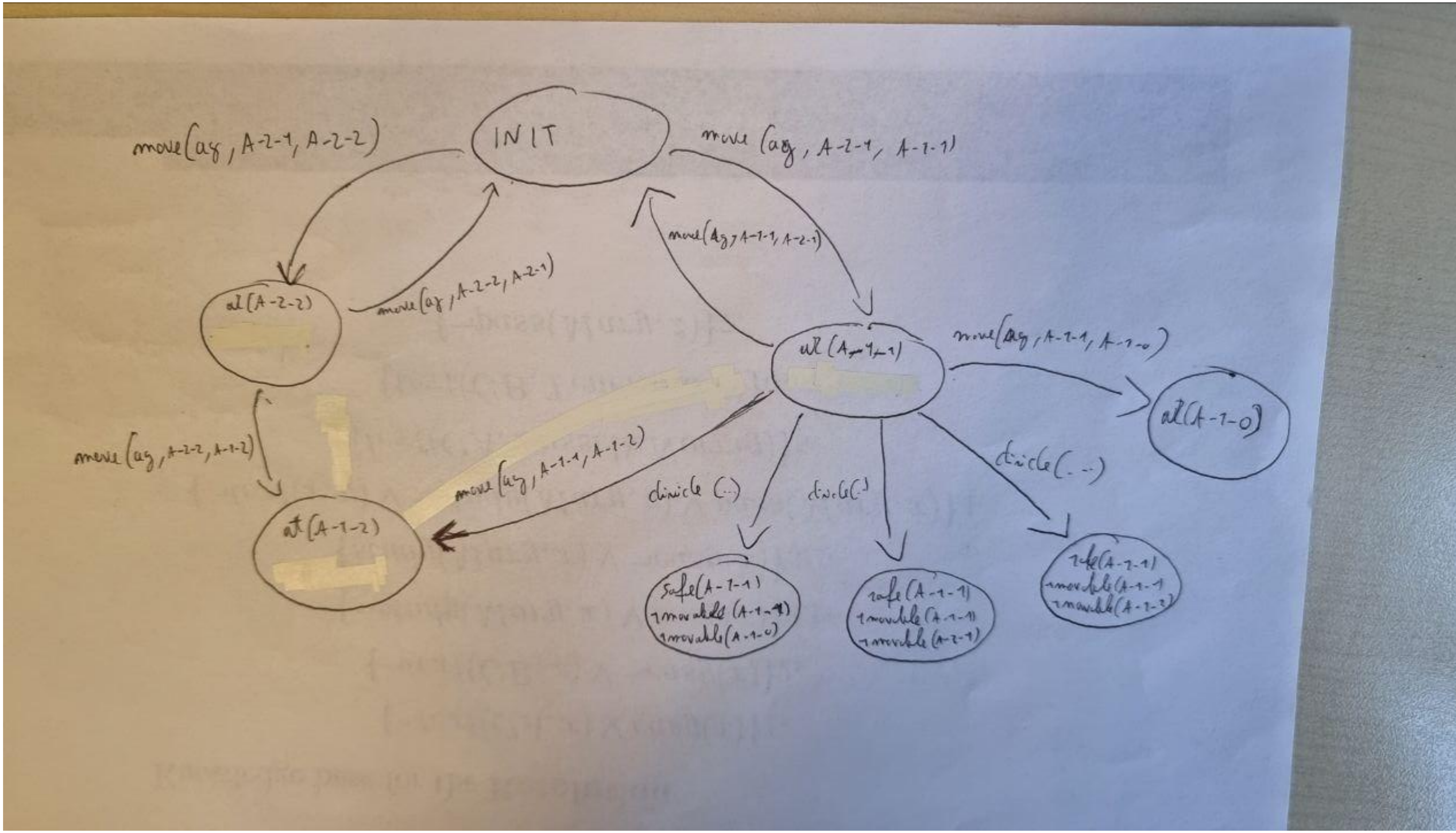
Exercise 1

```
(define (problem safe-party-problem)
        (:domain safe-party-domain)
        (:objects a g1 g2 A_0_0 B_0_0 A_0_1 B_0_1 A_0_2 B_0_2
        A_1_0 B_1_0 A_1_1 B_1_1 A_1_2 B_1_2
        A_2_0 B_2_0 A_2_1 B_2_1 A_2_2 B_2_2)
        (:init (ag a)(group g1)(group g2)
        (at a A_2_1)(at g1 A_1_1)(at g2 B_0_2)
        (movable A_0_0)(movable A_1_0)
        (movable A_1_1)(movable A_2_1)
        (movable A_0_2)(movable A_1_2)(movable A_2_2)
        (movable B_0_0)(movable B_2_0)
        (movable B_0_1)(movable B_1_1)(movable B_2_1)
        (movable B_0_2)(movable B_1_2)
```

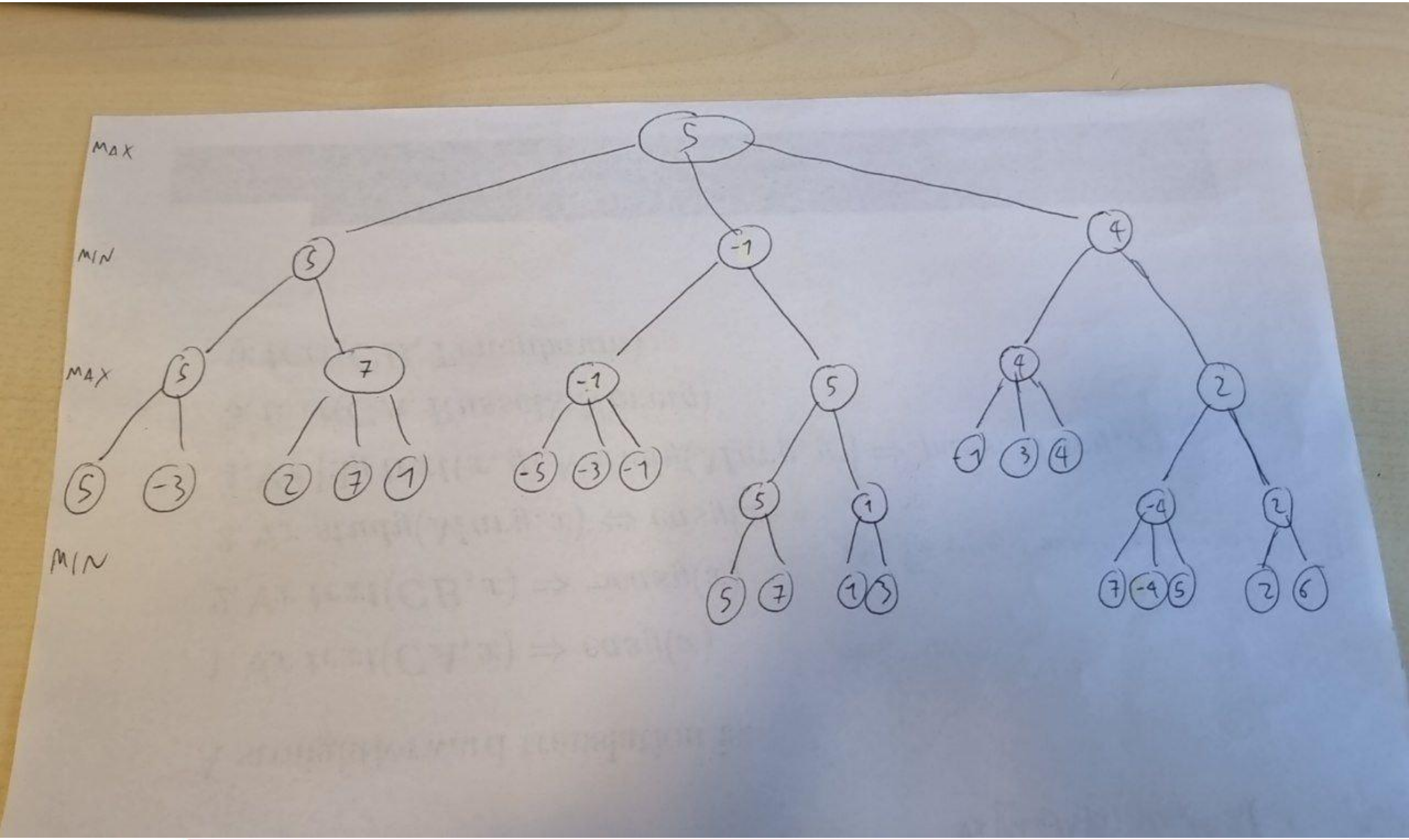
Exercise 1

```
(adj A_0_0 B_0_0)(adj B_0_0 A_0_0)
          (adj A_0_0 A_0_1)(adj A_0_0 A_1_0)
          (adj A_0_1 A_0_0)(adj A_0_1 A_0_2)(adj A_0_1 A_1_1)
          (adj A_0_2 A_0_1)(adj A_0_2 A_1_2)
          (adj A_1_0 A_0_0)(adj A_1_0 A_2_0)(adj A_1_0 A_1_1)
          (adj A_1_1 A_1_0)(adj A_1_1 A_0_1)(adj A_1_1 A_2_1)(adj A_1_1 A_1_2)
          (adj A_1_2 A_1_1)(adj A_1_2 A_0_2)(adj A_1_2 A_2_2)
          (adj A_2_0 A_1_0)(adj A_2_0 A_2_1)
          (adj A_2_1 A_2_0)(adj A_2_1 A_1_1)(adj A_2_1 A_2_2)
          (adj A_2_2 A_2_1)(adj A_2_2 A_1_2)
          (adj B_0_0 B_0_1)(adj B_0_0 B_1_0)
          (adj B_0_1 B_0_0)(adj B_0_1 B_0_2)(adj B_0_1 B_1_1)
          (adj B_0_2 B_0_1)(adj B_0_2 B_1_2)
          (adj B_1_0 B_0_0)(adj B_1_0 B_2_0)(adj B_1_0 B_1_1)
          (adj B_1_1 B_1_0)(adj B_1_1 B_0_1)(adj B_1_1 B_2_1)(adj B_1_1 B_1_2)
          (adj B_1_2 B_1_1)(adj B_1_2 B_0_2)(adj B_1_2 B_2_2)
          (adj B_2_0 B_1_0)(adj B_2_0 B_2_1)
          (adj B_2_1 B_2_0)(adj B_2_1 B_1_1)(adj B_2_1 B_2_2)
          (adj B_2_2 B_2_1)(adj B_2_2 B_1_2)
          (safe B_0_0)(safe B_1_0)(safe B_2_0)
          (safe B_0_1)(safe B_1_1)(safe B_2_1)
          (safe B_1_2)(safe B_2_2)
          (safe A_0_0)(safe A_1_0)(safe A_2_0)
          (safe A_0_1)(safe A_2_1)
          (safe A_0_2)(safe A_1_2)(safe A_2_2)
          )
          (:goal (and(safe A_1_1)(safe B_0_2)))
)
```

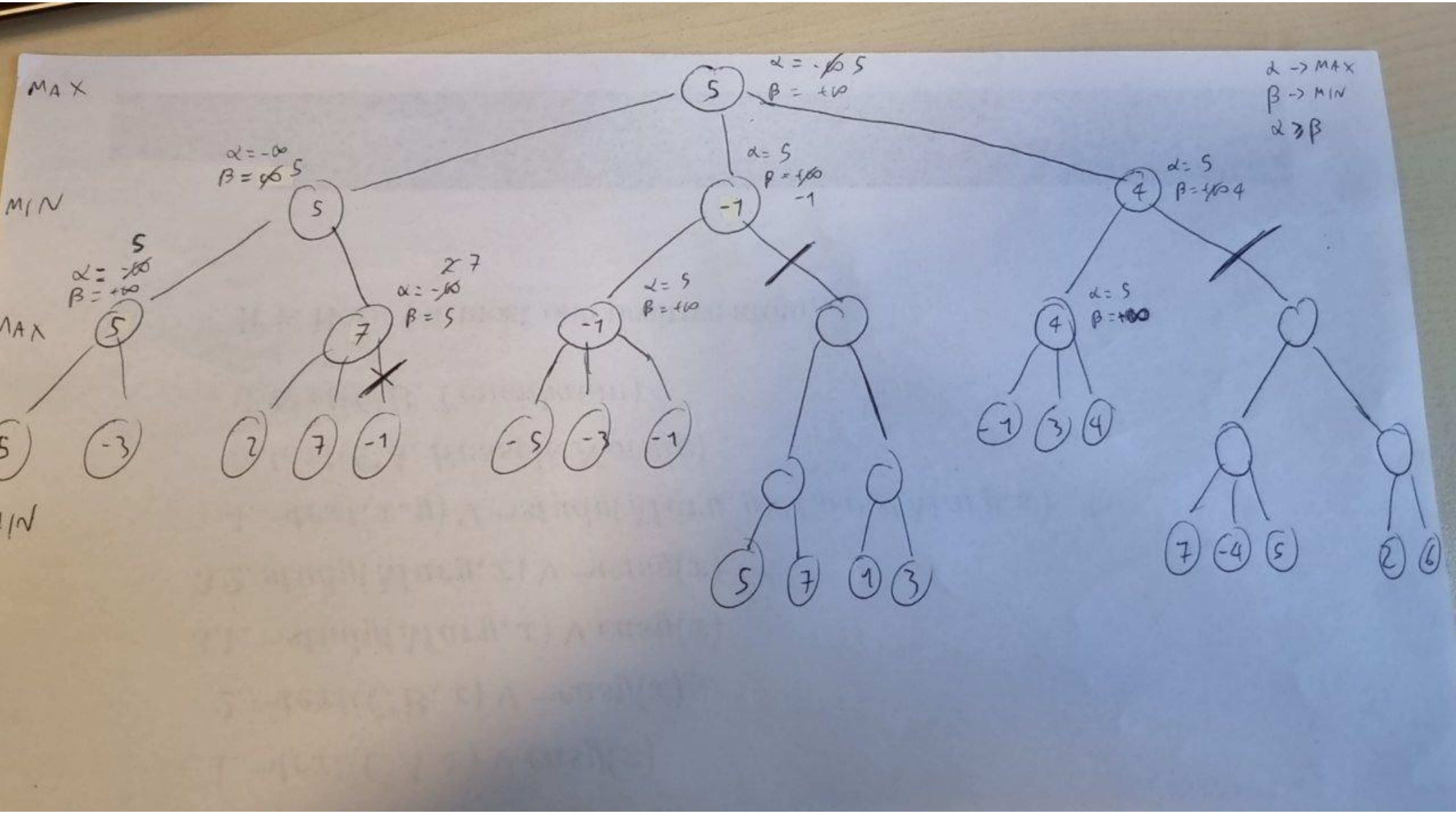
Exercise 1


INIT
move(ag, A-2-1, A-2-2)
move (ag, A-2-1, A-1-1)
move(ag, A-1-1, A-2-1)
at(A-2-2)
move(ag, A-2-2, A-2-1)
at (A-1-1)
move(ag, A-1-1, A-1-0)
at(A-1-0)
move (ag, A-2-2, A-1-2)
move (ag, A-1-1, A-1-2)
at(A-1-2)
safe(A-1-1)
safe(A-1-1)
safe(A-1-1)

Exercise 1



Exercise 2



Exercise 2

A possible reordering is, in the second subtree from left to right, to put 7 as the first node, in this way alpha becomes immediately greater than beta (5) and thus allowing us to prune all the nodes left in that subtree.

Exercise 3

i. Eliminate $\leftrightarrow$:
$$\neg\Big(\big(((A \wedge B) \rightarrow C) \wedge (A \vee B \vee C)\big) \rightarrow \big(((A \rightarrow B) \wedge (B \rightarrow A)) \rightarrow C\big)\Big)$$

ii. Eliminate $\rightarrow$:
$$\neg\Big(\neg\big((\neg(A \wedge B) \vee C) \wedge (A \vee B \vee C)\big) \vee \big(\neg((\neg A \vee B) \wedge (\neg B \vee A)) \vee C\big)\Big)$$

iii. Move $\neg$ inwards:
$$\big((\neg(A \wedge B) \vee C) \wedge (A \vee B \vee C)\big) \wedge \neg\big(\neg((\neg A \vee B) \wedge (\neg B \vee A)) \vee C\big) =$$
$$\big((\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C)\big) \wedge \big(((\neg A \vee B) \wedge (\neg B \vee A)) \wedge \neg C\big) =$$
$$(\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C) \wedge (\neg A \vee B) \wedge (\neg B \vee A) \wedge \neg C$$

CNF: $(\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C) \wedge (\neg A \vee B) \wedge (\neg B \vee A) \wedge \neg C$

Exercise 3

$\{\neg A, \neg D\}$ 1

$\{\neg A, D\}$ 2

$\{B, C, \neg D\}$ 3

$\{A, B\}$ 4

$\{\neg C, \neg D\}$ 5

$\{A, \neg B, \neg D\}$ 6

$\{A, \neg B, D\}$ 7

1 e 6 $\rightarrow$ $\{\neg B, \neg D\}$ 8

4 e 7 $\rightarrow$ $\{A, D\}$ 9

2 e 9 $\rightarrow$ $\{D\}$ 10

8 e 10 $\rightarrow$ $\{\neg B\}$ 11

3 e 5 $\rightarrow$ $\{B, \neg D\}$ 12

10 e 12 $\rightarrow$ $\{B\}$ 13

11 e 13 $\rightarrow$ $\{\}$

Exercise 3

i. Eliminate $\leftrightarrow$:
$\neg\Big(\big(((A \wedge B) \rightarrow C) \wedge (A \vee B \vee C)\big) \rightarrow \big(((A \rightarrow B) \wedge (B \rightarrow A)) \rightarrow C\big)\Big)$

ii. Eliminate $\rightarrow$:
$\neg\Big(\neg\big((\neg(A \wedge B) \vee C) \wedge (A \vee B \vee C)\big) \vee \big(\neg((\neg A \vee B) \wedge (\neg B \vee A)) \vee C\big)\Big)$

iii. Move $\neg$ inwards:
$\big((\neg(A \wedge B) \vee C) \wedge (A \vee B \vee C)\big) \wedge \neg\big(\neg((\neg A \vee B) \wedge (\neg B \vee A)) \vee C\big) =$
$\big((\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C)\big) \wedge \big(((\neg A \vee B) \wedge (\neg B \vee A)) \wedge \neg C\big) =$
$(\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C) \wedge (\neg A \vee B) \wedge (\neg B \vee A) \wedge \neg C$

CNF: $(\neg A \vee \neg B \vee C) \wedge (A \vee B \vee C) \wedge (\neg A \vee B) \wedge (\neg B \vee A) \wedge \neg C$

Exercise 3

i. Unit propagation: $B \mapsto F$
$\{\{A, C, D\}, \{\neg C, \neg D\}, \{C, \neg D\}, \{A, \neg C\}\}$

ii. Splitting rule: $A \mapsto F$
$\{\{C, D\}, \{\neg C, \neg D\}, \{C, \neg D\}, \{\neg C\}\}$

iii. Unit propagation: $C \mapsto F$
$\{\{D\}, \{\neg D\}\}$

iv. Unit propagation: $D \mapsto T$
$\{\square\}$

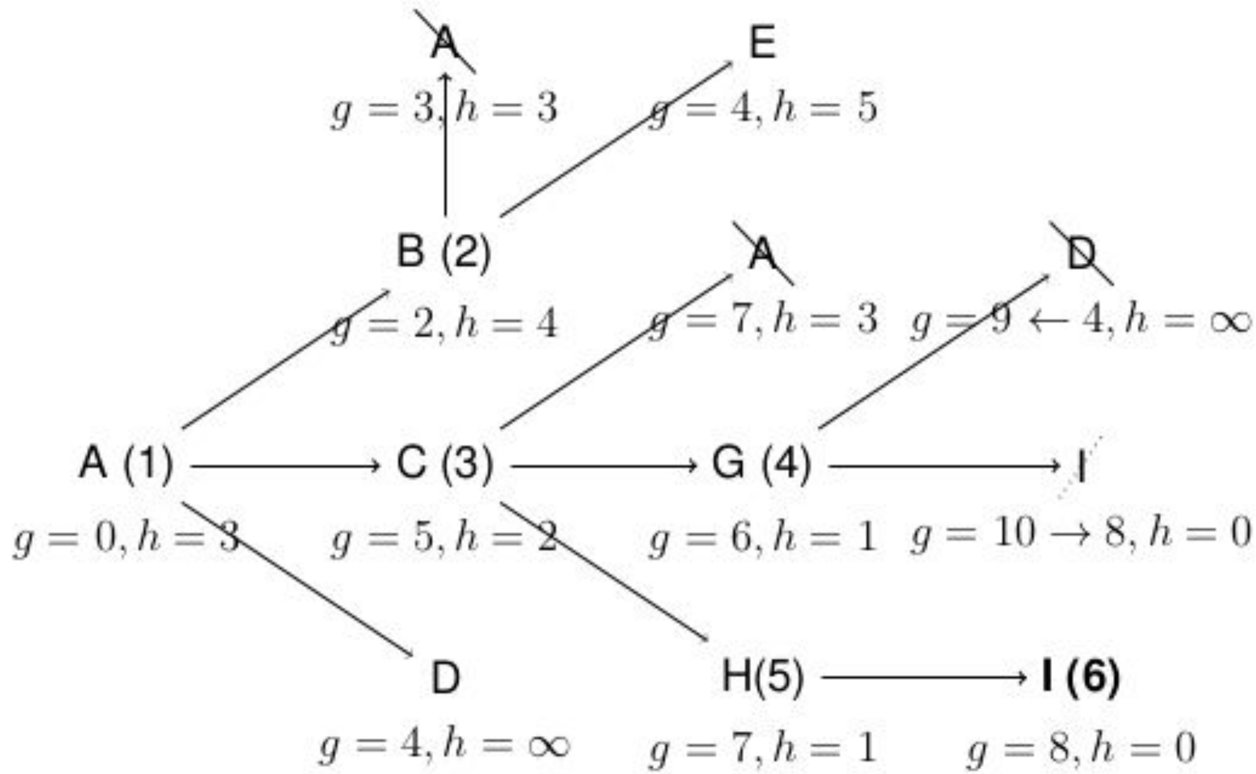
v. Backtracking: $A \mapsto T$
$\{\{\neg C, \neg D\}, \{C, \neg D\}, \}$

vi. Splitting rule: $C \mapsto F$
$\{\{\neg D\}\}$

vii. Unit propagation: $D \mapsto F$
$\{\}$

Satisfying assignment: $A, \neg B, \neg C, \neg D$.

Exercise 4


A
$g = 3, h = 3$
E
$g = 4, h = 5$
B (2)
$g = 2, h = 4$
A
$g = 7, h = 3$
D
$g = 9 \leftarrow 4, h = \infty$
A (1)
$g = 0, h = 3$
C (3)
$g = 5, h = 2$
G (4)
$g = 6, h = 1$
I
$g = 10 \rightarrow 8, h = 0$
D
$g = 4, h = \infty$
H(5)
$g = 7, h = 1$
I (6)
$g = 8, h = 0$

Exercise 4

Hill climbing returns the solution A, C, G, I, which is not optimal. The expansion order is as follows:

- State A
  $A \rightarrow B$: $h(B) = 4$
  $A \rightarrow C$: $h(C) = 2 \Leftarrow$ expand
  $A \rightarrow D$: $h(D) = \infty$
- State C
  $C \rightarrow A$: $h(A) = 3$
  $C \rightarrow G$: $h(G) = 1 \Leftarrow$ expand
  $C \rightarrow H$: $h(H) = 1$
- State G
  $G \rightarrow D$: $h(D) = \infty$
  $G \rightarrow I$: $h(I) = 0 \Leftarrow$ expand
- State I
  $I \rightarrow F$: $h(F) = 6$
- I is a local minimum, search stops.

Exercise 4

Yes, it is possible that hill-climbing stops without finding a solution. Consider the following heuristic:

State s	A	B	C	D	E	F	G	H	I
h(s)	2	2	2	2	2	2	2	2	0

Hill climbing starts at A (with $h(A) = 2$) and chooses randomly one of its child nodes (B, C, or D). Since either one of them has an equal heuristic value of 2, Hill Climbing stays at node A, which is then returned as a local minimum without finding the solution.