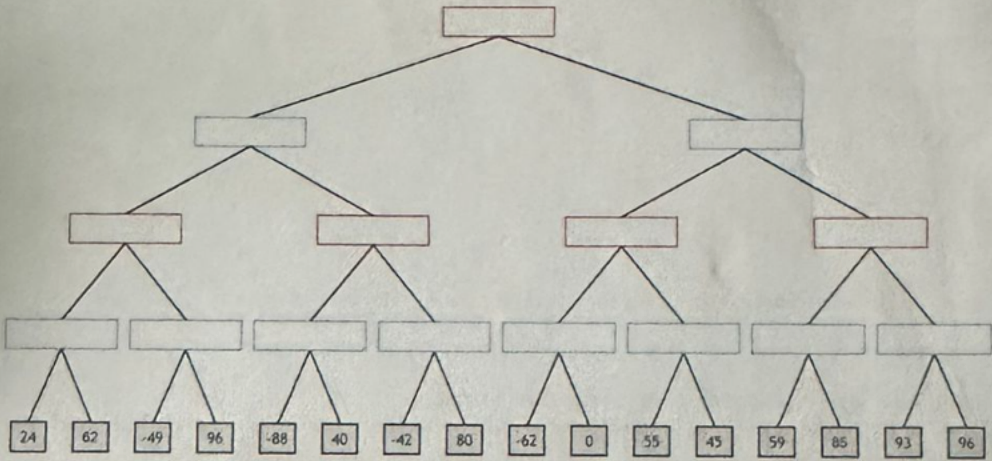
EXAM OF FUNDAMENTALS OF AI – FIRST MODULE
SIMULATION OF EXAM

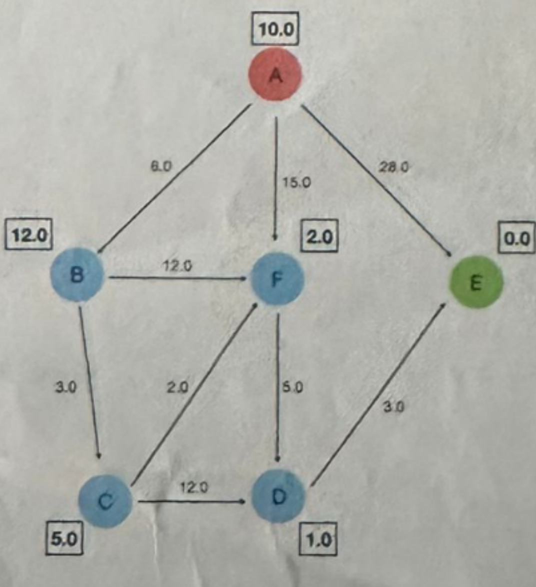
Exercise 1

Consider the following game tree where the first player is *MAX*. Show how the *min-max* algorithm works and show the *alfa-beta* cuts. Also, show which is the proposed move for the first player.



Exercise 2

Consider the following graph, where A is the initial node and E the goal node. The number on each arc is the cost of the operator for the move, while the number in the square next to each node is the heuristic evaluation of the node itself, namely, its estimated distance from the goal.



a) Apply the depth-first search (do not consider the costs of the nodes), and draw the developed search tree indicating the expansion order; in the case of non-determinism, choose the nodes to expand according to the alphabetical order. What is the produced solution and its cost?

b) Apply search A*, and draw the developed search tree indicating the expansion order and the value of the function f(n) for each node n. In the case of non-determinism, choose the nodes to expand according to the alphabetical order. Consider as heuristic h(n) the one indicated in the square next to each node in the figure. What is the produced solution and its cost?

Exercise 3

Given the following CSP:

A, B, C::[0..7]

A + 1 <= B
A + 4 >= C
B + 3 <= C

Apply the Arc-consistency to the CSP and show the final domains of the three variables.

Exercise 4

In the initial state described by the following atomic formulas:
[handempty, in(keys, pocket), in(letter, mailbox), entrance(door, home), compatible(keys, door)]

You want to reach the goal:
have(letter), inside(home)

The actions are modeled as follows:

grab(Object, Container)
PRECOND: handempty, in(Object, Container), ¬inside(home)
DELETE: handempty, in(Object, Container)
ADD: have(Object)

open(Entrance)
PRECOND: compatible(Key, Entrance), have(Key), ¬open(Entrance)
DELETE: have(Key)
ADD: handempty, open(Entrance)

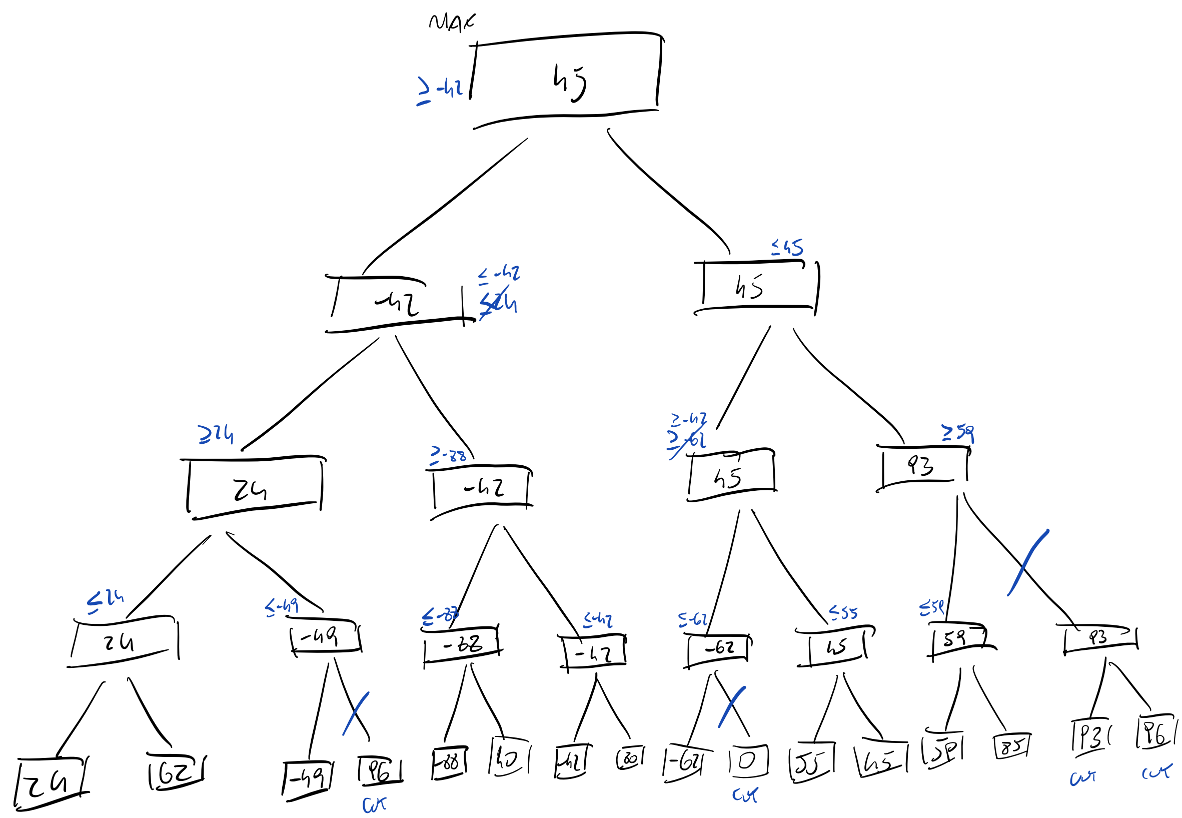
get_inside(Place)
PRECOND: ¬inside(Place), open(Entrance), entrance(Entrance, Place)
DELETE: open(Entrance)
ADD: inside(Place)

Solve the problem with the POP algorithm, identifying threats and their solution during the process.

Exercise 5

1) Model the action **open** (preconditions, effects and frame axioms), and the initial state of the exercise 4 using the Kowalsky formulation.
2) Show two levels of graph plan when applied to exercise 4.
3) What is Breadth-First Search? Describe this search strategy and discuss its completeness and complexity
4) What are the main features of a local search algorithm?
5) What are the main approaches of deductive planning. Explain the main differences.

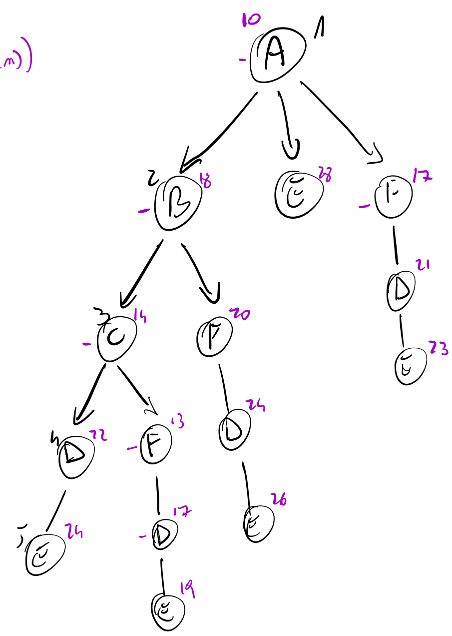
1)



2)

A^*: $f(n)$ in purple $(h(n)+c(n))$

NODE	COST	
A	10	
A F	17	
A B	18	
A BC	14	
A BCF	13	
A BCFD	17	
A BCFDE	19	← GOAL

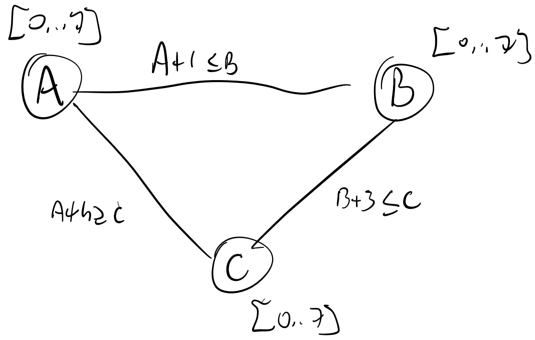


DF:

A
B E F
C F E F
D F F E F
E F F E F
. GOAL

Solution: A B C D E
Cost: 6+3+12+3=24

3)



1° iteration

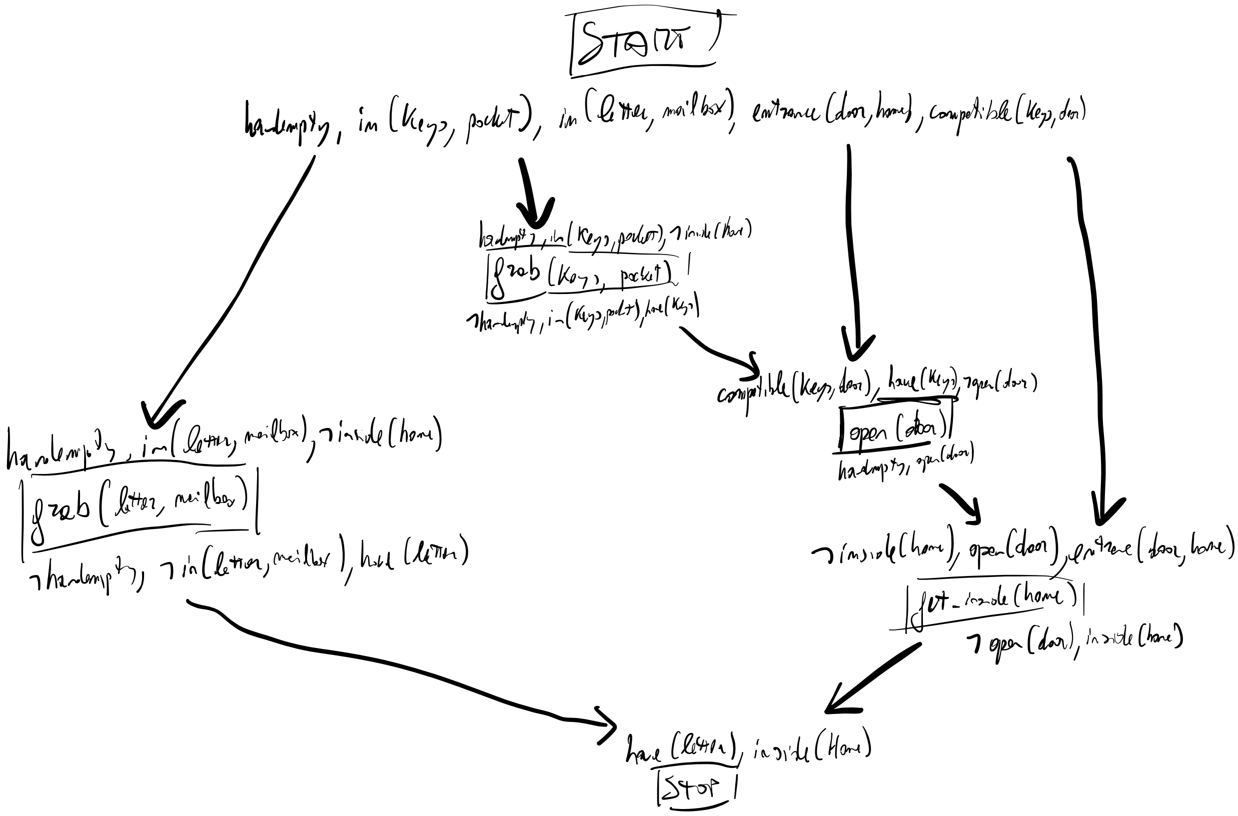
			CURRENT A	AFTER B	STEP C
	A	B			
$A+1 \le B$	$[0,1,2,3,4,5,6,\cancel{7}]$	$[\cancel{0},1,2,3,4,5,6,7]$	$[0...6]$	$[1...7]$	$[0...7]$
	B	C			
$B+3 \le C$	$[1,2,3,4,\cancel{5},\cancel{6},\cancel{7}]$	$[\cancel{0},\cancel{1},\cancel{2},\cancel{3},4,5,6,7]$	$[0...6]$	$[1...4]$	$[4...7]$
	A	C			
$A+4 \ge C$	$[0,1,2,3,4,5,6]$	$[4,5,6,7]$	$[0...6]$	$[1...4]$	$[4...7]$

2° iteration

			CURRENT A	AFTER B	STEP C
	A	B			
$A+1 \le B$	$[0,1,2,3,4,\cancel{5},\cancel{6}]$	$[1,2,3,4]$	$[0...4]$	$[1...4]$	$[4...7]$
	B	C			
$B+3 \le C$	$[1,2,3,4]$	$[4,5,6,7]$		satisfied	
	A	C			
$A+4 \ge C$	$[0,1,2,3,4]$	$[4,5,6,7]$		satisfied	

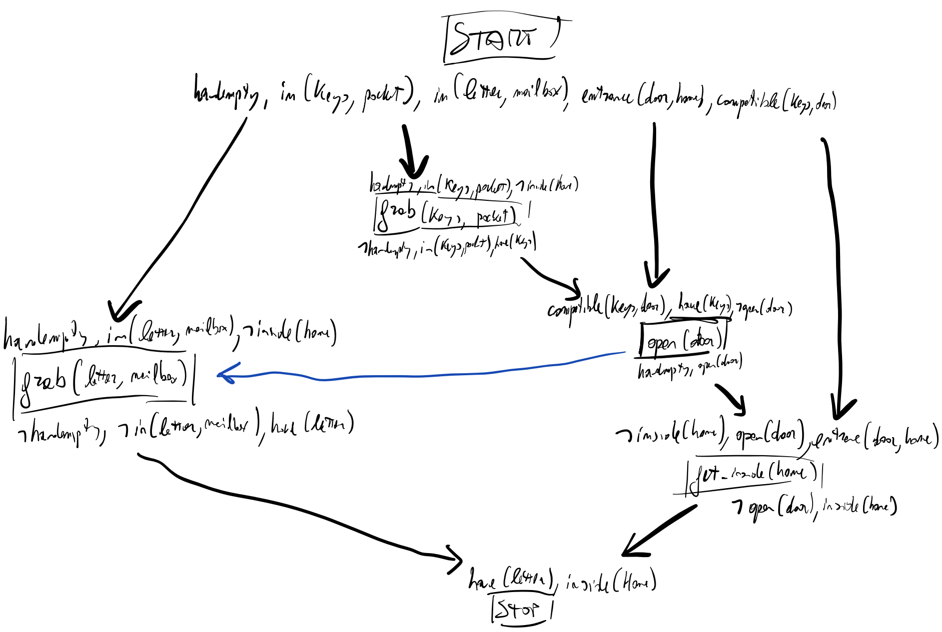
	A	B	C
All constraint are satisfied, final domain	$[0...4]$	$[1...4]$	$[4...7]$

b)



Threats:

- grab(letter, mailbox), grab(keys, pocket) interfere on handempty, so an order constraint is needed. Since open(door) restore handempty, the arrow could be added for this edge.
- point on CWA (theory)



grab(keys, pocket)
open(door)
grab(letter, mailbox)
get_inside(home)

5.1)

holds(handempty, s0).

holds(in(keys, pocket), s0).

holds(in(letter, mailbox), s0).

holds(entrance(door, home), s0).

holds(compatible(keys, door), s0).

holds(handempty, do(open(entrance), S)).

holds(open(entrance), do(open(entrance), S)).

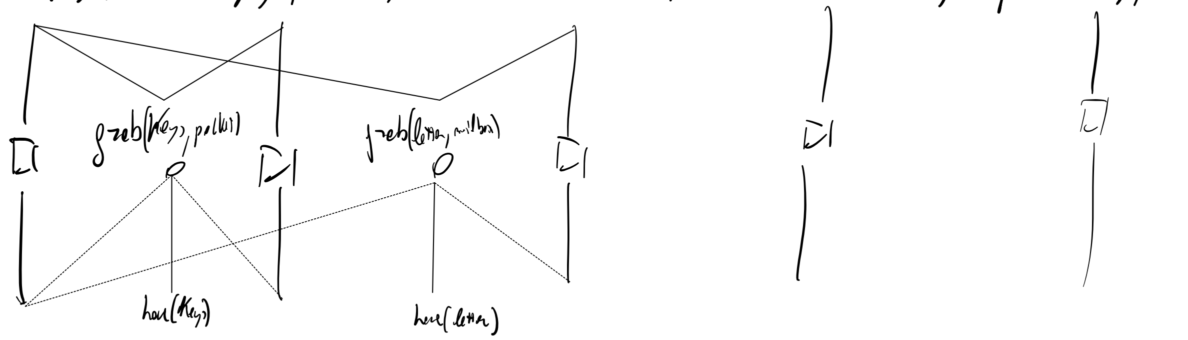
poss(open(Entrance), S) :- holds(compatible(Key, entrance), S), holds(have(Key), S), holds(V, S), $V \neq$ inside(home)

holds(V, do(open(Entrance), S)) :- holds(V, S), $V \neq$ have(Key).

$\square$ no-op

5.2)

handempty, in(keys, pocket), in(letter, mailbox), entrance(door, home), compatible(keys, door)



inconsistency:

[no-op(handempty)
  grab(keys, pocket)

[no-op(in(keys, pocket))
  grab(keys, pocket)

[no-op(handempty)
  grab(letter, mailbox)

[no-op(in(letter, mailbox))
  grab(letter, mailbox)

[handempty
  have(keys)

[in(keys, pocket)
  have(keys)

[handempty
  have(letter)

[in(letter, mailbox)
  have(letter)