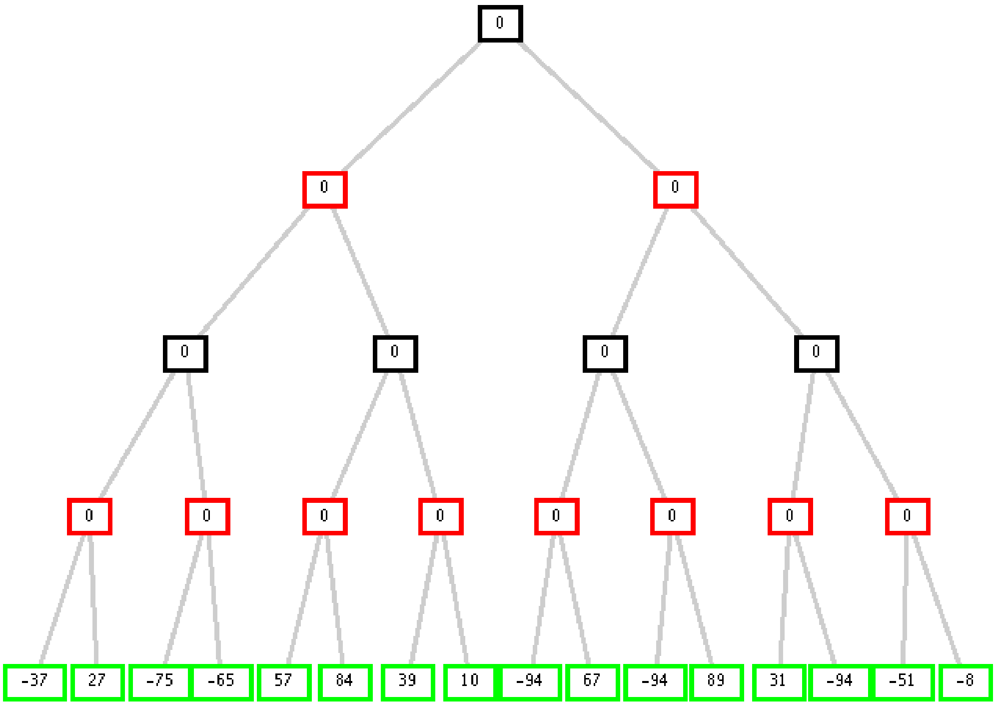
EXAM OF FUNDAMENTALS OF AI – FIRST MODULE
23/12/2021 – T2
PROF. MICHELA MILANO

Exercise 1

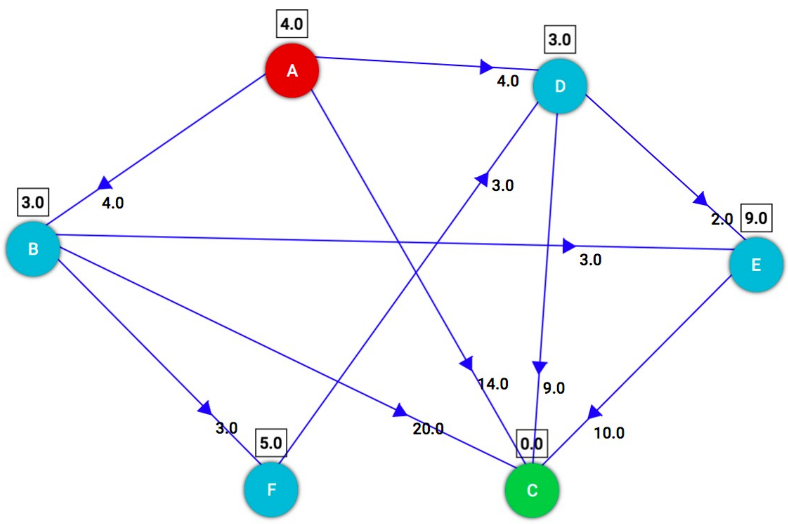
Consider the following game tree where the first player is MAX. *Show how the* min-max algorithm works and show the *alfa-beta* cuts. Also, show which is the proposed move for the first player.



Exercise 2

Consider the following graph, where A is the initial node and C the goal node, and the number associated with the arcs is the cost of the operator to go from the starting node to the arrival node of the arc. Next to each node, in a square, the heuristic estimate of its distance from the goal node is also indicated.

a) Show the search tree generated by the A* algorithm along with the order of expansion of nodes. In case of ties, chose the node to expand in alphabetical order.
b) Is the heuristic admissible?
c) Which is the cost of the path found by A* and the number of nodes expanded?



Exercise 3
Consider the 4 Queens Problem (4 queens on a 4x4 board to avoid attacks). Remember that queens attack horizontally, vertically and diagonally.
Represent the problem as a CSP by modeling each variable/queen with a column and considering the domain values given by the possible rows (1,..., 4) in which to place the queens. Represent variables, domains and constraints in which they are involved.

X1, X2, X3, X4 :: [1,2,3,4]

X1 =/= X2 =/=X3=/=X4

|Xi-Xj| =/= |i-j|

Then solve the problem by applying the Forward Checking (FC) technique. Consider the variables giving priority to those representing columns with lower index. For the choice of domain values, consider those with a lower value. We highlight the domains and how they change at each step, together with the backtracking points.

Then solve the problem again by applying the Partial Look Ahead (PLA) at each step after FC. What changes regarding backtracking?

Exercise 4
Given the following initial state **[at(room1), handempty, available_paint, near(room1,room2), near(room2,room1)]**: and actions modeled as follows:

paint(Loc)
PRECOND: available_paint, at(Loc), robot_has_brush
DELETE: available_paint
ADD: painted(Loc)

putdown_brush
PRECOND: robot_has_brush
DELETE: robot_has_brush
ADD: handempty

pickup_brush
PRECOND: handempty
DELETE: handempty
ADD: robot_has_brush

charge_paint
PRECOND: not available_paint
DELETE: -
ADD available_paint

go(Loc1, Loc2)
PRECOND: at(Loc1), near(Loc1,Loc2)
DELETE: at(Loc1)
ADD at(Loc2)

and the following goal **painted(room1), painted(room2)**

Solve the problem by using the POP algorithm showing threats and how to solve them.

Exercise 5

1) Model the action **coloring** (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 are the main approaches of deductive planning. Explain the main differences.

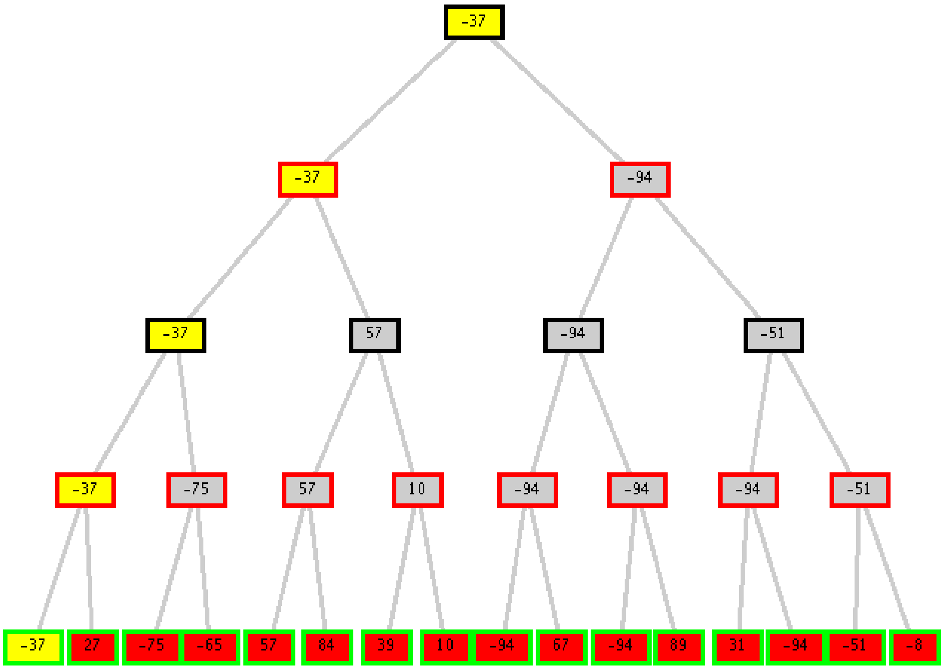
4) What are metaheuristics? Describe the main algorithms that have been presented during the course.

5) What is arc-consistency? Describe the algorithm to achieve it. Explain the properties of values that are removed from constraints and of values that are left in the domains.

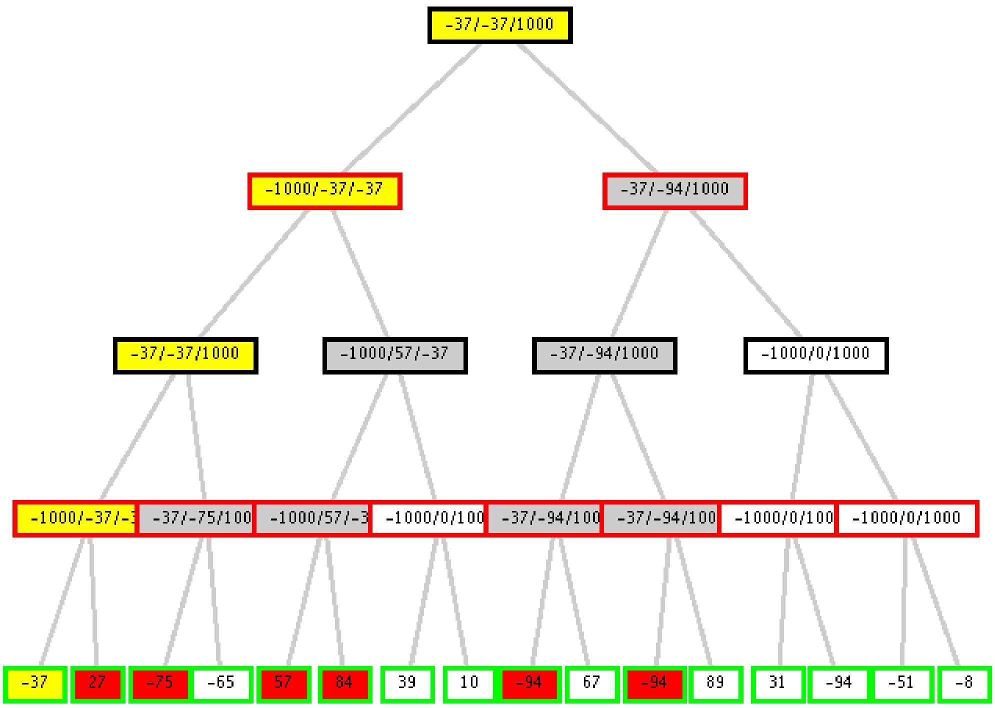
Solution

Exercise 1

Min-max

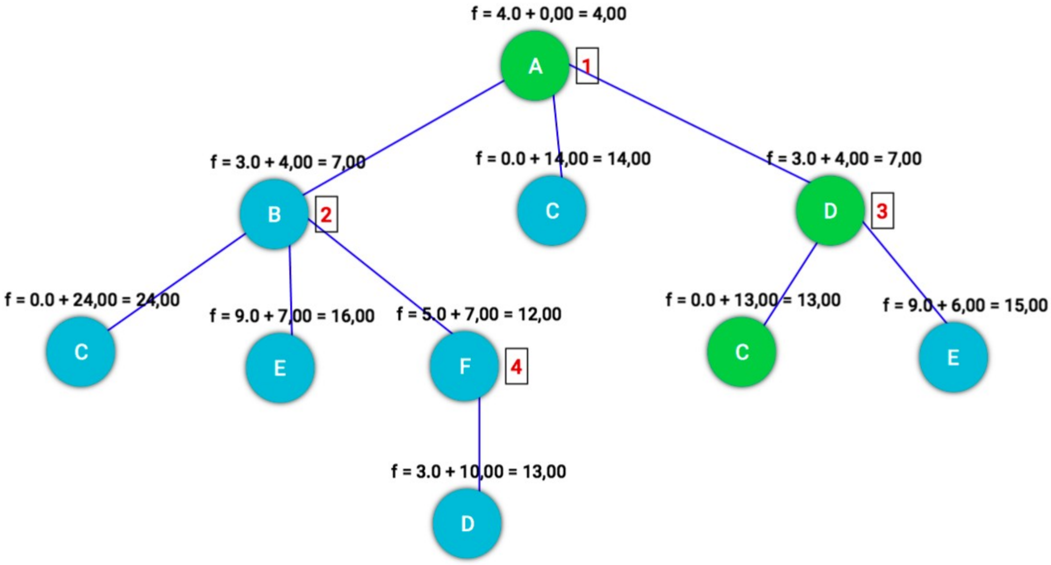


Alfa-beta :



Exercise 2

A*: L’euristica è ammissibile.
Con A*, costo cammino trovato (ADC in verde) pari a 13 (cammino ottimo)



Exercise 3

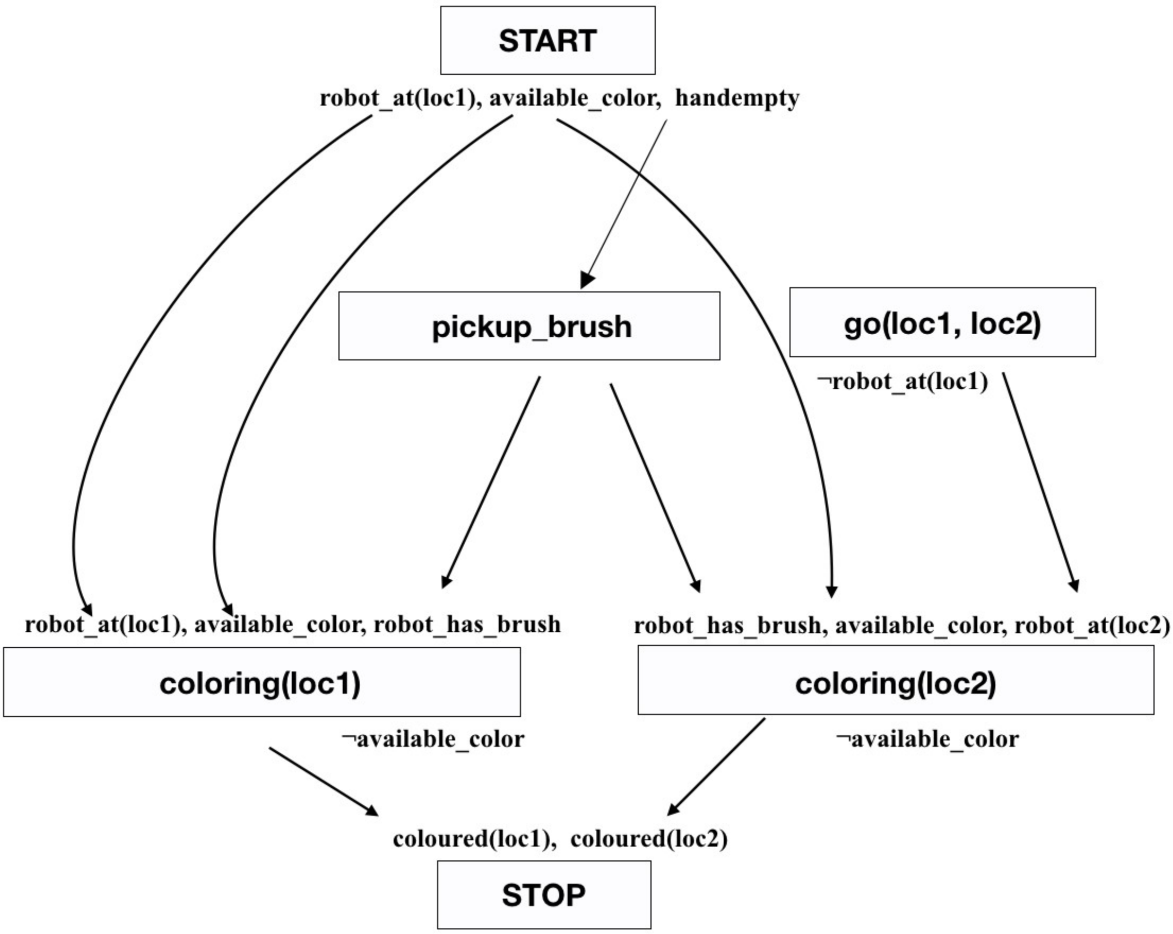
Solo Forward Checking (2 backtraking effettuati)

X1	X2	X3	X4	
[1..4]	[1..4]	[1..4]	[1..4]	
1	[3,4]	[2,4]	[2,3]	
1	3	[]	[2]	fallimento dominio X3 vuoto; backtraking cronologico
1	4	[2]	[3]	
1	4	2	[]	fallimento dominio X4 vuoto; backtraking cronologico
2	[4]	[1,3]	[1,3,4]	
2	4	[1]	[1,3]	
2	4	1	[3]	
2	4	1	3	

PLA: (1 solo backtraking svolto)

X1	X2	X3	X4	
[1..4]	[1..4]	[1..4]	[1..4]	
1	[3,4]	[2,4]	[2,3]	dopo Forward checking
1	[4]	[4]	[2,3]	dopo PLA
1	4	[]	[3]	dopo Forward checking; backtraking cronologico
2	[4]	[1,3]	[1,3,4]	dopo forward checking
2	[4]	[1,3]	[1,3,4]	dopo PLA
2	4	[1]	[1,3]	dopo FC
2	4	[1]	[1,3]	dopo PLA
2	4	1	[3]	dopo FC (e PLA)
2	4	1	3	

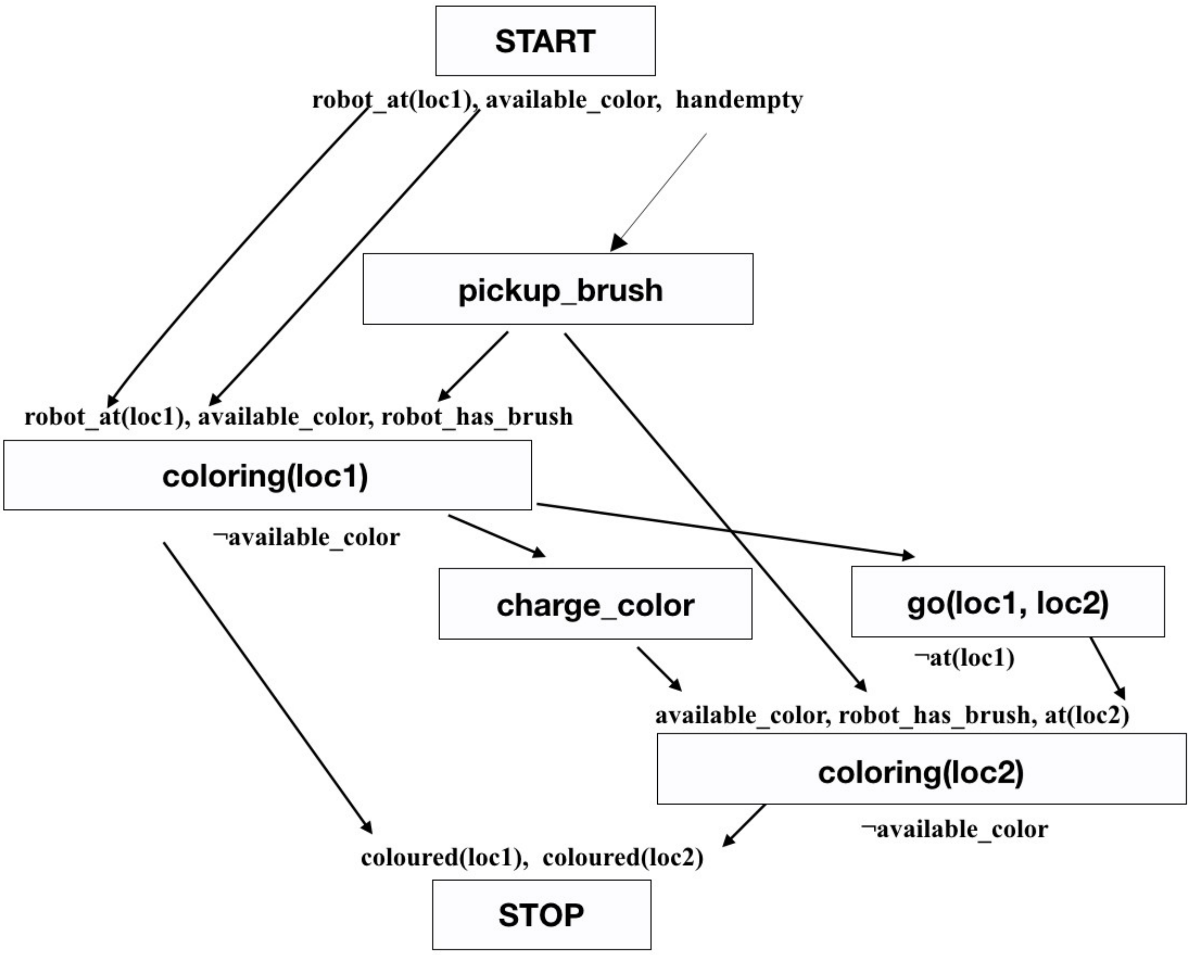
Exercise 4



The plan up to now contains threats. In particular:
<Start, coloring(loc2), available_color> and <Start, coloring(loc1), available_color> are threatened by coloring(loc1) and coloring(loc2) respectively. No ordering constraints can solve these threats: we need to insert a white knight charge_color.

In addition the action go(loc1, loc2) threats causal link <Start, coloring(loc1), at(loc1)>

In this case demotion can solve the threat. We introduce an ordering constraint between coloring(loc1) and go(loc1, loc2).



Note that we have to remove the causal link between the start and the

coloring(loc2) for have charge and insert the new causal link between charge_color and coloring(loc2) for available_color.

Exercise 5

1)

holds(at(loc1),s0).
holds(available_color, s0).
holds(handempty,s0).
holds(colored(loc), do(coloring(loc),S))
pact(coloring(loc),S):- holds(available_color, S), holds(at(loc),S), holds(robot_has_brush,S).
holds(V,do(coloring(loc),S*)):- holds(V,S), V\=available_color.*

2)