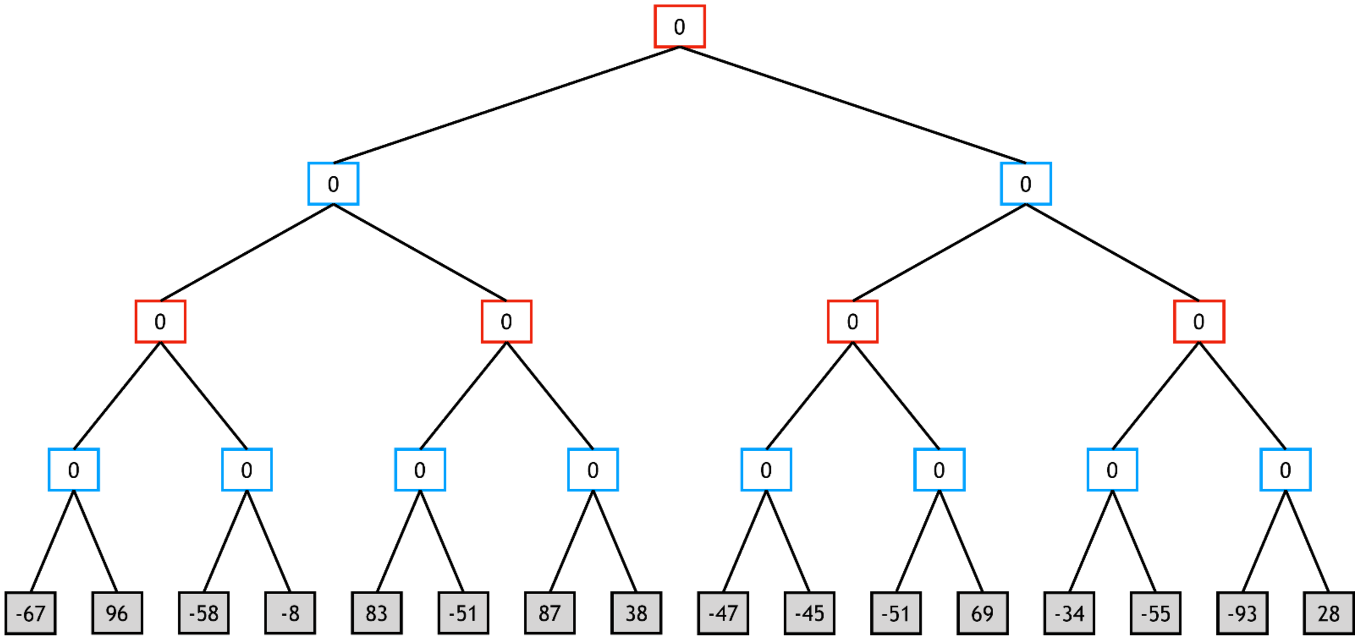
EXAM OF FUNDAMENTALS OF AI – FIRST MODULE
26/01/2023
PROF. MICHELA MILANO

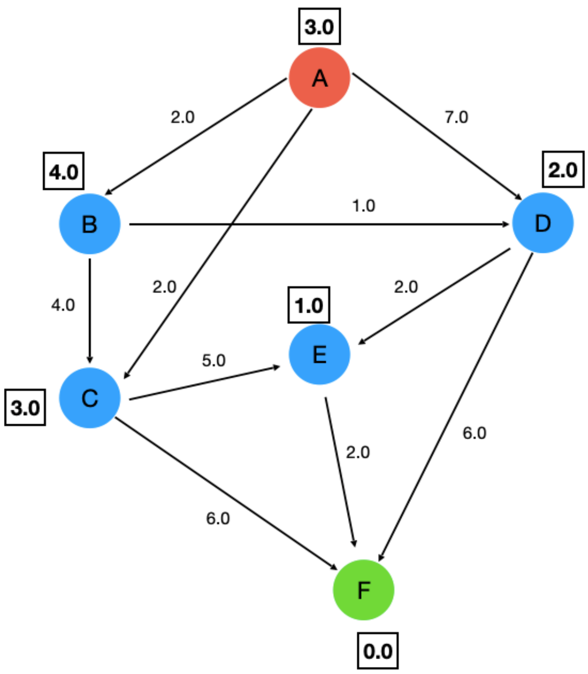
Exercise 1

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 F 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::[1, 2, 3, 4, 5]
B::[1, 2, 3, 4, 5]
C::[1, 2, 3, 4, 5]
D::[1, 2, 3, 4, 5]

$A > 1 + B$
$B < D - 2$
$A = D + 1$
$C = B - 1$

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

Exercise 4

Given the following initial state:
at(risorgimento), has(badge)

Given the following actions:
take_exam_at(Location)
PRECOND: at(Location), has(badge)
DELETE: -
ADD: exam_done_at(Location)

swap(Object1, Object2)
PRECOND: has(Object1)
DELETE: has(Object1)
ADD: has(Object2)

take_bus(LocationA, LocationB)
PRECOND: at(LocationA), has(ticket)
DELETE: at(LocationA)
ADD: at(LocationB)

We have to reach the goal:
exam_done_at(risorgimento), exam_done_at(zamboni)

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

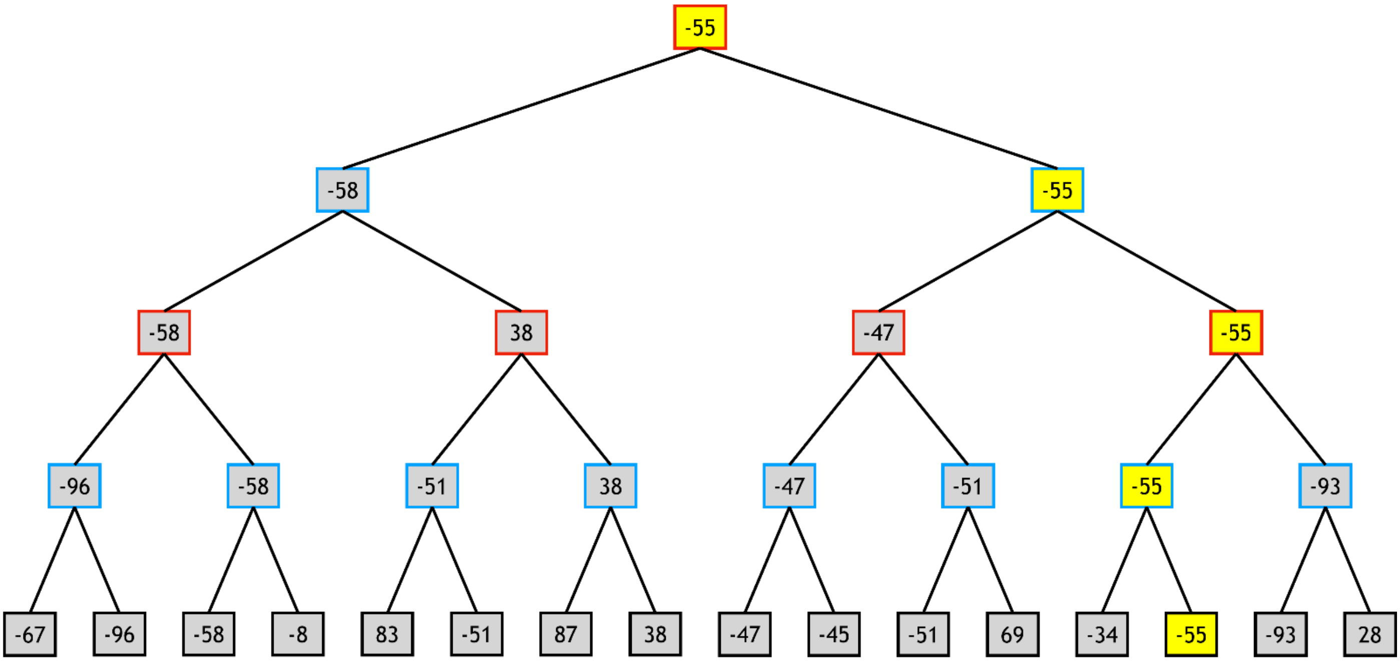
Exercise 5

1) Model the action **swap** (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.

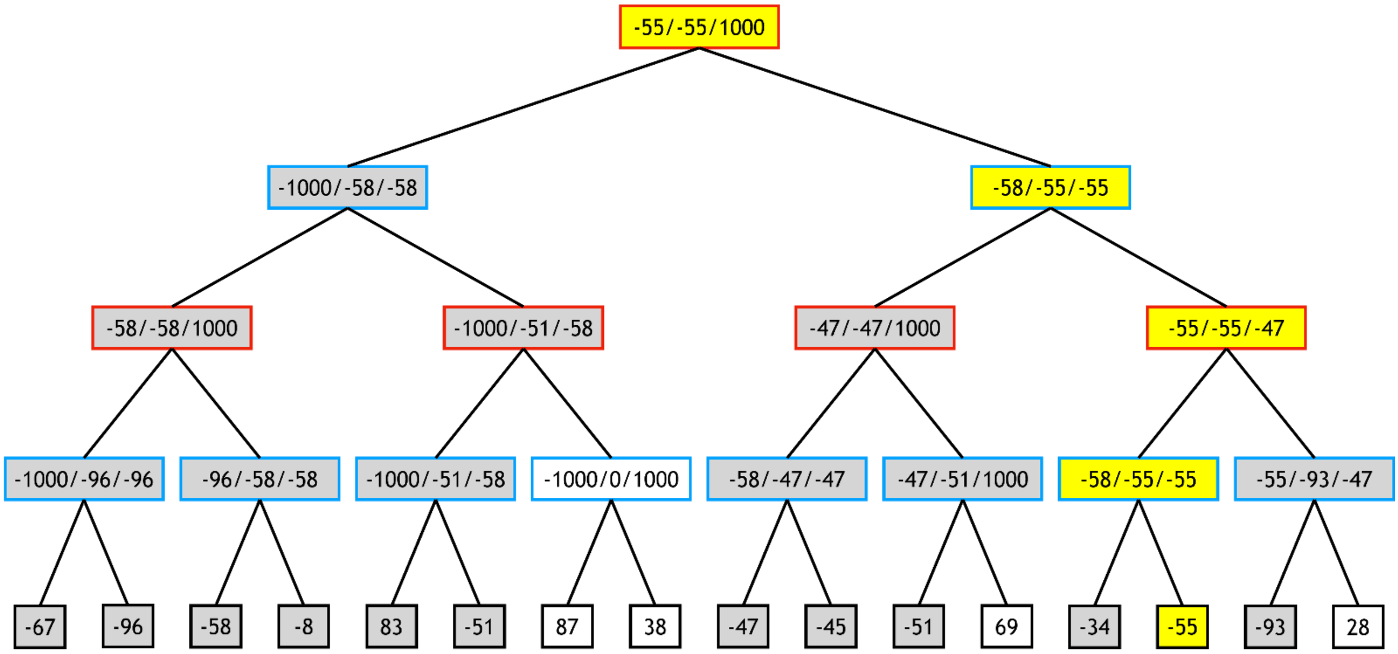
SOLUTION

Exercise 1

a) Min-max

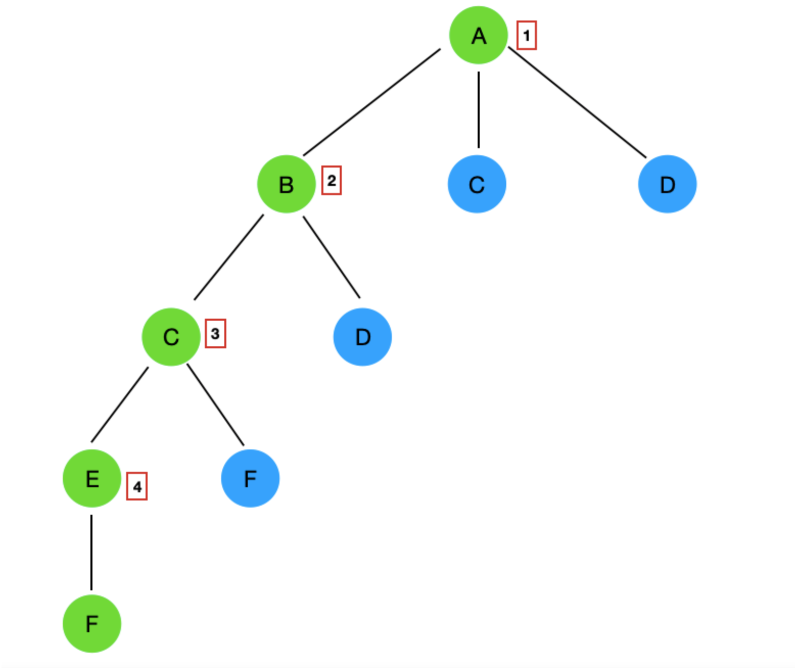


b) Alpha-beta cuts



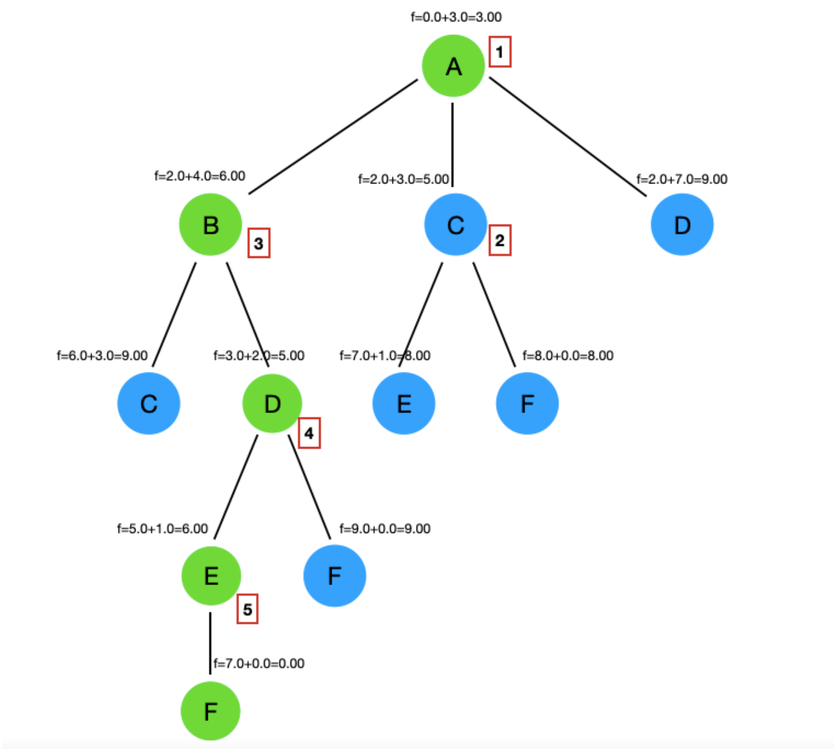
Exercise 2

a) Depth-first search



The solution is ABCEF with cost 13.

b) A*



The produced solution is ABDEF with cost 7

Exercise 3

***Applied reasoning*:**

We start with the following domains:
A::[1, 2, 3, 4, 5]
B::[1, 2, 3, 4, 5]
C::[1, 2, 3, 4, 5]
D::[1, 2, 3, 4, 5]

And with the following set of constraints:
a) A > 1 + B
b) B < D - 2
c) A = D + 1
d) C = B – 1

First Iteration:

Apply (a)
- For each A, is there a B? No. The only values of A which admit a solution are [3, 4, 5]
- For each B, is there an A? No. The only values of B which admit a solution are [1, 2, 3]

New domains:
A::[3, 4, 5]
B::[1, 2, 3]
C::[1, 2, 3, 4, 5]
D::[1, 2, 3, 4, 5]

Apply (b)
- For each B, is there a D? No. The only values of B which admit a solution are [1, 2]
- For each D, is there a B? No. The only values of D which admit a solution are [4, 5]

New domains:
A::[3, 4, 5]
B::[1, 2]
C::[1, 2, 3, 4, 5]
D::[4, 5]

Apply (c)
- For each A, is there a D? No. The only values of A which admit a solution are [5]
- For each D, is there a A? No. The only values of D which admit a solution are [4]

New domains:
A::[5]
B::[1, 2]
C::[1, 2, 3, 4, 5]
D::[4]

Apply (d)
- For each B, is there a C? No. The only values of B which admit a solution are [2]
- For each C, is there a B? No. The only values of C which admit a solution are [1]

New domains:
A::[5]
B::[2]
C::[1]
D::[4]

Second Iteration (since the domains have changed during the last one):

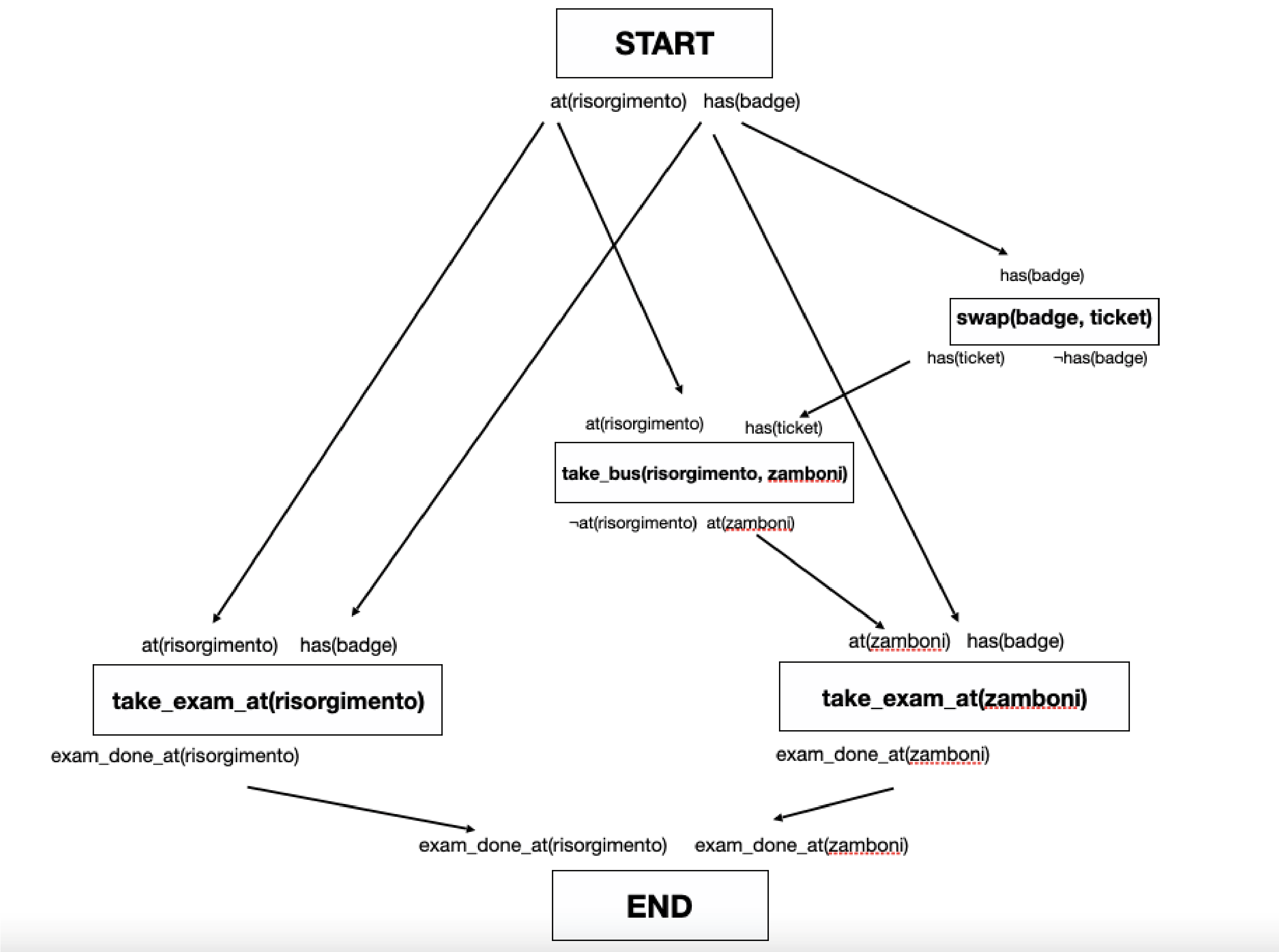
Apply (a)
- For each A, is there a B? Yes.
- For each B, is there an A? Yes.

Domains unchanged.

Apply (b)
- For each B, is there a D? No. There is no assignment for B which could satisfy the constraint.

Algorithm termination. The problem is infeasible.

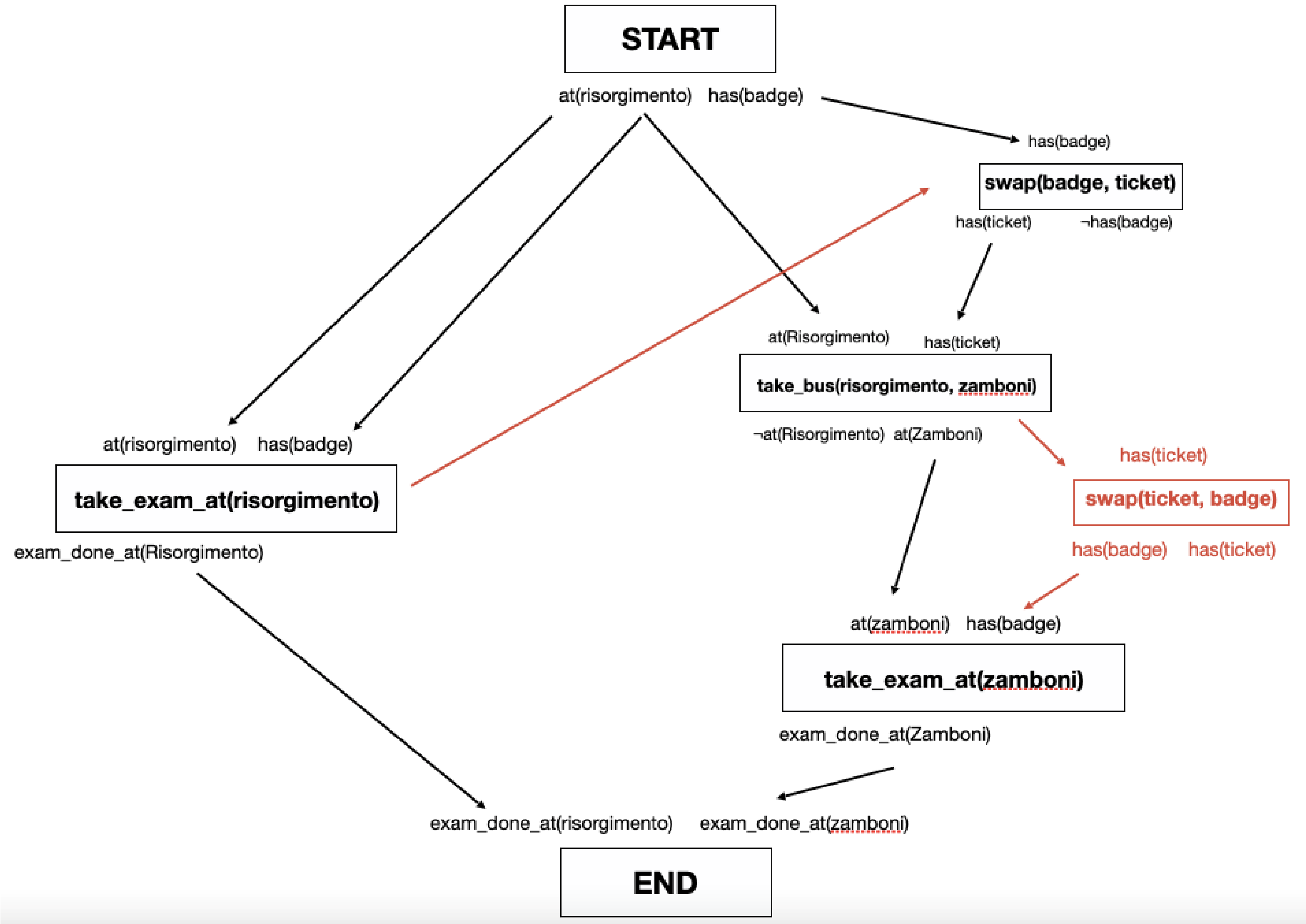
Exercise 4



This plan contains threats. In particular:

- The action *swap(badge, ticket)* threats causal link <*start*, *take_exam_at(risorgimento)*, *has(badge)*>. In this case, demotion can solve the threat, therefore we introduce an ordering constraint between *take_exam_at(risorgimento)* and *swap(badge, ticket)*. The same ordering constraint can be used to solve the threat to the causal link <*start*, *take_exam_at(risorgimento)*, *at(risorgimento)*> which is introduced by the action *take_bus(risorgimento, zamboni)*.
- The action *swap(badge, ticket)* threats <*start*, *take_exam_at(zamboni)*, *has(badge)*>. No ordering constraint can solve this threat, therefore we need to insert a white knight *swap(ticket, badge)*.

The final planning is:



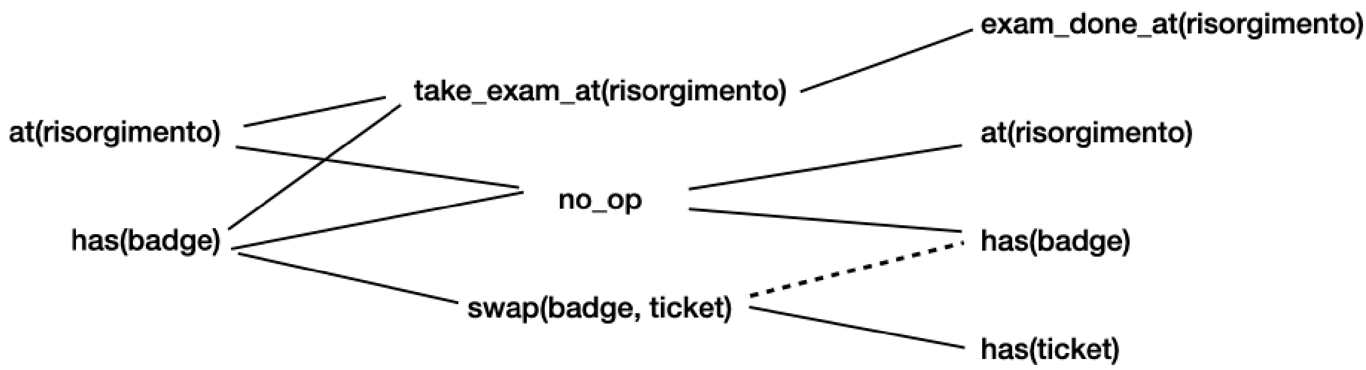
Exercise 5

1) Kowalsky formulation of action swap.

Initial State
holds(at(risorgimento), s0)
holds(has(badge),s0)

Action: swap(object1, object2)
holds(has(Object2), do(swap(Object1, Object2), S))
pact(swap(Object1, Object2), S):- holds(has(Object1), S)
holds(V, do(swap(Object1, Object2), S)):- holds(V, S), V\=has(Object1)

2) Graphplan



no_op on **has(badge)** and **swap(badge, ticket)** are incompatible

has(badge) and **has(ticket)** are incompatible