AERO (Automated Escape Room Operations)

Sapienza Università di Roma

Planning and Reasoning - 2025/2026

Matricola: Federico De Lullo 1935510

Project Description

The goal is to efficiently coordinate the escape of an agent from a high-security Escape Room under strict resource **constraints**.

The agent interacts with the environment to solve logical puzzles, requiring specific sequences of actions to unlock containment units.

Throughout the process, it must navigate the layout while balancing energy consumption and time limits against movement **costs**.

Finally, to successfully evacuate, the system must ensure sufficient residual resources remain to activate the exit mechanism

PDDL

Three Different Scenarios:

1. Pure Logic
2. Battery Constraint
3. Time and Battery Constraint

Scenario 1 - Predicates

Predicates	Description
(has ?k - key)	Indicates that the agent currently possesses the specified key
(open ?c - container)	Indicates that a container is currently open and its contents are accessible
(locked ?c - container)	Indicates that a container is locked and requires a specific key to be opened.
(in ?k - key ?c - container)	Specifies that a specific key is located inside a specific container
(on ?s - switch)	Indicates that an electronic switch has been activated
(escaped)	The goal state; becomes true when the agent successfully leaves the room.
(fits ?k - key ?c - container)	Defines the logical relationship: which specific key is capable of opening which container
(is-master-key ?k - key)	Identifies the unique key required to trigger the final escape mechanism

Scenario 1 - Actions

Action	Preconditions	Effects
PICK-UP	• The key (?k) is **not inside** any container.	• The agent **has** the key (?k)
TAKE-KEY	• The container (?c) is **open** • The key (?k) is **inside** the container (?c)	• The agent **has** the key (?k) • The key is **no longer inside** the container
UNLOCK	• The agent **has** the key (?k) • The key **fits** the container (?c) • The container is **locked**	• The container becomes **unlocked** • The container becomes **open**
TOGGLE-SWITCH	• The switch (?s) is currently **OFF** (not on).	• The switch becomes **ON** (active)
ESCAPE	• The agent **has** the **Master Key** • **ALL** switches are **ON**	• The agent **escapes** • **Goal** reached

Scenario 1 – Initial Situation

Initial Situation
(locked box1)
(locked box2)
(fits k-start box1)
(fits k-inter box2)
(is-master-key k-master)
(in k-inter box1)
(in k-master box2)

Scenario 1 – Result with Fast Downward

Metrics	Blind Search	H-Max
Plan Cost	10	10
Plan Length	10	10
Search Time	0,0036s	0,0042s
Total Time	0,0158s	0,0163s
Expanded Nodes	97	89
Evaluated Nodes	97	97
Generated Nodes	417	393
Dead Ends	0	0

Scenario 1 – Plans Found

Blind Search	H-Max
pick-up-free k-start	*pick-up-free k-start*
toggle-switch sw4	*unlock k-start box1*
toggle-switch sw3	*take-key k-inter box1*
toggle-switch sw2	*unlock k-inter box2*
toggle-switch sw1	*toggle-switch sw4*
unlock k-start box1	*toggle-switch sw3*
take-key k-inter box1	*toggle-switch sw2*
unlock k-inter box2	*toggle-switch sw1*
take-key k-master box2	*take-key k-master box2*
escape k-master	*escape k-master*

Scenario 2 – Predicates and Function

Predicate	Description
(at ?c - cell)	Tracks the current location of the agent within the grid.
(item_at ?i - item ?c - cell)	Indicates the specific cell where an item is currently located.
(holding ?i - item)	Indicates that the agent has picked up an item and is carrying it.
(exited)	The boolean goal state; becomes true only after the exit action is successfully performed
(adjacent ?c1 ?c2 - cell)	Defines the connectivity graph: which cells are directly connected to each other (allow movement)
(wall_between ?c1 ?c2 - cell)	Identifies a special connection blocked by an obstacle. Crossing this requires the high-cost **JUMP** action.
(is_exit ?c - cell)	Marks a specific cell as the designated escape point.

Function	Description
(battery) - number	Represents the agent's energy level. This value decreases with every action (move, jump, pick, exit) and constrains the feasibility of the plan.

Scenario 2 - Actions

Action	Precondition	Effect	Cost
MOVE	• Agent is at the starting cell. • Target cell is **adjacent**. • There is **NO wall** blocking the path. • Battery level must be at least **1**.	• Agent moves to the new cell. • Battery decreases by **1**.	**-1**
JUMP	• Agent is at the starting cell. • Target cell is **adjacent**. • There IS a **wall** between the cells. • Battery level must be at least **20** (High energy requirement).	• Agent jumps over the obstacle to the new cell • Battery decreases by **20**	**-20**
PICK	• Agent is at the same location as the item. • Battery level must be at least **1**	• Agent holds the item. • Item is removed from the floor. • Battery decreases by **1**.	**-1**
EXIT	• Agent is at the designated **Exit Cell**. • Agent is holding the **Key Card**. • Battery level must be at least **5** (Reserve energy).	• Agent successfully leaves the maze. • **Goal State** achieved.	**N/A** *(Requires 5)*

Scenario 2 – Results with ENHSP

Metrics	H-Add	H-Max
Plan Length	27	33
Planning Time	104ms	76ms
Heuristic Time	20ms	27ms
Search Time	59ms	69ms
Expanded Nodes	51	296
Evaluated Nodes	106	368
Duplicates Nodes	20	258
Dead Ends	0	38

Scenario 2 – Plans Found

Step	Sat-Hadd (Length: 27)	Opt-Hmax (Length: 33)
0	jump c1_1 c1_2	jump c1_1 c1_2
1	move c1_2 c1_3	move c1_2 c1_3
2	**move c1_3 c1_2**	move c1_3 c2_3
3	**move c1_2 c1_3**	move c2_3 c3_3
4	move c1_3 c2_3	move c3_3 c3_4
5	move c2_3 c3_3	move c3_4 c2_4
6	move c3_3 c3_4	move c2_4 c2_5
7	move c3_4 c2_4	**move c2_5 c2_4**
8	move c2_4 c2_5	**move c2_4 c2_5**
9	move c2_5 c3_5	**move c2_5 c2_4**
10	move c3_5 c4_5	**move c2_4 c2_5**
11	move c4_5 c5_5	**move c2_5 c2_4**
12	move c5_5 c6_5	**move c2_4 c2_5**
13	move c6_5 c6_6	**move c2_5 c2_4**
14	pick key_card c6_6	**move c2_4 c2_5**
15	move c6_6 c6_5	move c2_5 c3_5
16	move c6_5 c5_5	move c3_5 c4_5
17	move c5_5 c4_5	move c4_5 c5_5
18	move c4_5 c3_5	move c5_5 c6_5
19	move c3_5 c2_5	move c6_5 c6_6
20	move c2_5 c2_4	pick key_card c6_6
21	move c2_4 c3_4	move c6_6 c6_5
22	move c3_4 c3_3	move c6_5 c5_5
23	move c3_3 c2_3	move c5_5 c4_5
24	move c2_3 c1_3	move c4_5 c3_5
25	move c1_3 c1_2	move c3_5 c2_5
26	exit c1_2 key_card	move c2_5 c2_4
27		move c2_4 c3_4
28		move c3_4 c3_3
29		move c3_3 c2_3
30		move c2_3 c1_3
31		move c1_3 c1_2
32		exit c1_2 key_card

Scenario 3 – Predicates and Functions

Predicate	Description
(at ?l - location)	Tracks the agent's current position in the network.
(active ?s - sensor)	Indicates whether a specific sensor has been successfully activated. The goal requires **ALL** sensors to be active.
(exited)	The final state achieved only after successfully leaving from the Room.
(connected ?l1 ?l2 - location)	Defines the structure of the world
(is_charger ?l - location)	Identifies the special location where the recharge action can be performed.
(sensor_at ?s - sensor ?l - location)	Maps each sensor to its specific module location.

Function	Description
(battery) - number	Represents energy. Consumed by movement and activation. It can be restored to 100% via the recharge action.
(time_left) - number	Represents the mission deadline. Consumed by **every** action (including moving, working, and recharging). Unlike battery, **Time cannot be replenished**

Scenario 3 - Actions

Action	Precondition	Effect	Cost
MOVE	• Agent is at starting location. • Locations are **connected**. • **Battery ≥ 10**. • **Time ≥ 5**.	• Agent changes location.	**-10 Battery** **-5 Time**
ACTIVATE	• Agent is at the sensor's location. • Sensor is currently **inactive**. • **Battery ≥ 15** • **Time ≥ 10**	• Sensor becomes **active**.	**-10 Time**
RECHARGE	• Agent is at the **Hub** • **Battery < 90** • **Time ≥ 20**	• **Battery resets to 100%**. • Huge time penalty applied.	**-20 Time**
EXIT	• Agent is at the **Hub**. • **Time > 0** • **ALL sensors** are active	• Agent leaves the facility. • **Goal** reached.	**N/A** *(requires >0 Time)*

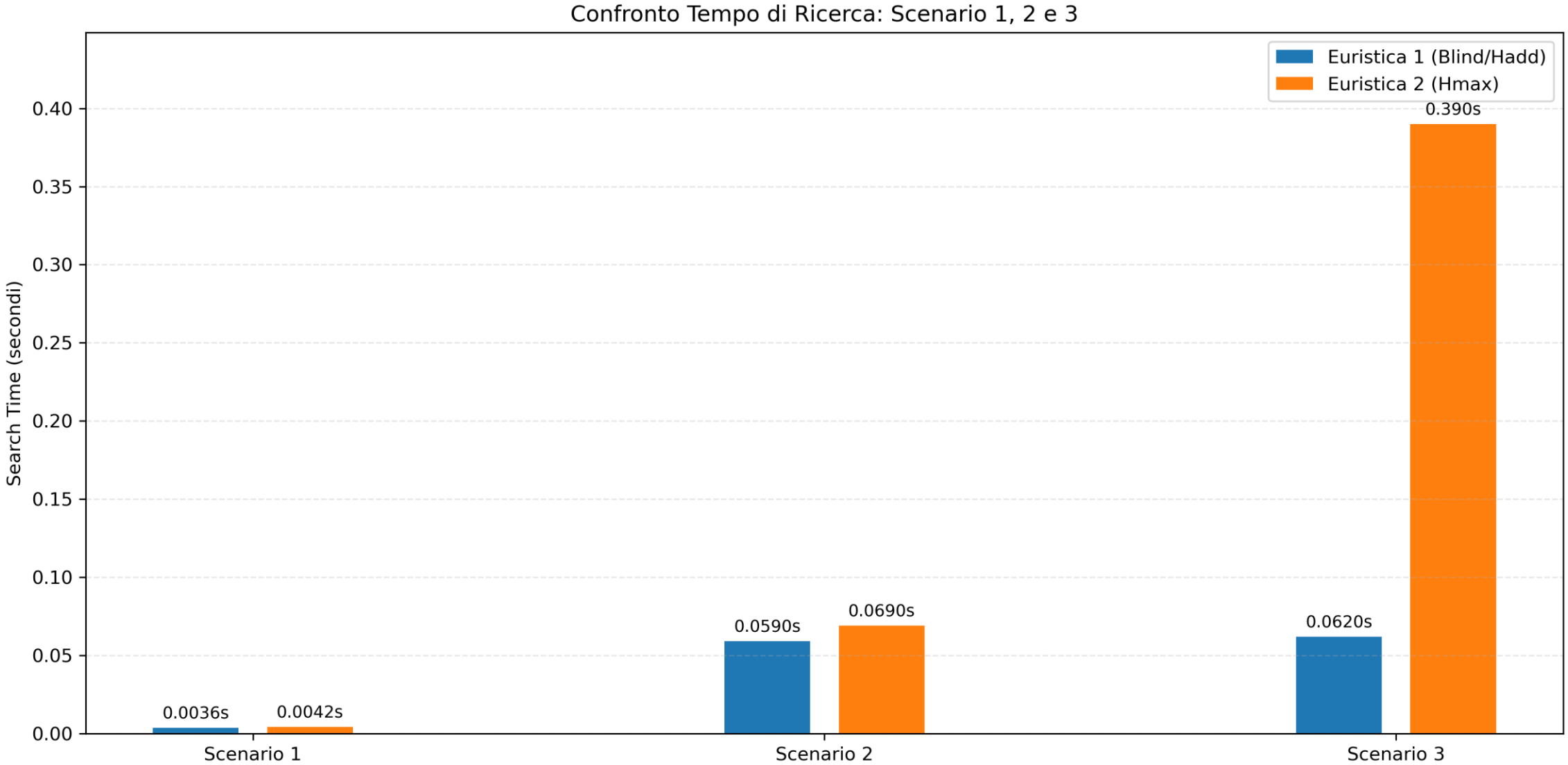
Scenario 3 – Results With EHNSP

Metrics	H-Add	H-Max
Plan Length	19	37
Planning Time	0.078s	0.406s
Search Time	0,062s	0,390s
Expanded Nodes	40	9465
Evaluated Nodes	186	24187
Dead Ends	9	10562

Scenario 3 - Actions

Step	Sat-Hadd (19 Steps)	Opt-Hmax (37 Steps)
0	move hub mod3	recharge hub
1	activate s3 mod3	move hub mod1
2	move mod3 hub	move mod1 hub
3	recharge hub	recharge hub
4	move hub mod6	move hub mod5
5	activate s6 mod6	move mod5 mod4
6	move mod6 mod5	move mod4 hub
7	activate s5 mod5	move hub mod3
8	move mod5 hub	move mod3 hub
9	recharge hub	move hub mod3
10	move hub mod1	activate s3 mod3
11	activate s1 mod1	move mod3 hub
12	move mod1 mod2	recharge hub
13	activate s2 mod2	move hub mod3
14	move mod2 hub	move mod3 hub
15	move hub mod4	move hub mod1
16	activate s4 mod4	activate s1 mod1
17	move mod4 hub	move mod1 hub
18	exit hub	move hub mod3
19		move mod3 hub
20		recharge hub
21		move hub mod3
22		move mod3 mod4
23		activate s4 mod4
24		move mod4 mod5
25		move mod5 hub
26		move hub mod3
27		move mod3 mod2
28		activate s2 mod2
29		move mod2 hub
30		recharge hub
31		move hub mod6
32		activate s6 mod6
33		move mod6 mod5
34		activate s5 mod5
35		move mod5 hub
36		exit hub

Comparison Among All Scenarios



IndiGolog

Controllers:

1. Reactive

Reasoning Task:

1. Legality Task
2. Projection Task

IndiGolog

IndiGolog is used to validate the domain logic and resource constraints before execution.

Legality Check:

Verifies if a specific sequence of actions is executable starting from the initial state.

Example: Trying to open_panel without has_key results in a failure.

Projection:

Predicts the future state of numerical resources (Battery & Time) after a sequence of actions.

Example: Checking if battery > 150 after activating systems confirms if the plan is safe.

IndiGolog – Relational Fluent

Fluent	Description
container_open	True if the secure container has been opened.
has_key	True if the agent holds the key required for the panel.
panel_open	True if the maintenance panel is accessible.
activated(S)	True if switch *S* (a, b, or c) is currently active.
calibrated(S)	True if sensor *S* is calibrated and working.
exited	True if the agent has successfully left the room.

IndiGolog – Functional Fluents

Fluent	Description
battery	Numeric value representing the remaining energy. Decreases with actions.
time_left	Numeric countdown representing the deadline. Decreases with every action.

IndiGolog – Reactive Controller

The **Reactive Controller** manages execution in a dynamic environment using prioritized_interrupts. It handles:

High Priority Emergencies:

If battery < 20, it immediately triggers a recharge action.

Exogenous Events (Faults):

If an external event jams a sensor (jam_sensor), the controller detects the neg(calibrated) state and re-calibrates it.

Context-Aware Execution:

Chooses between activate_auto (High Battery) and activate_manual (Low Battery) automatically.

Exogenous Actions	Internal Action	Effect
Leak	**battery_leak**	Instantly reduces battery level by 20 units.
jam1	**jam_sensor(s1)**	Forces Sensor 1 status to Uncalibrated.
jam2	**jam_sensor(s2)**	Forces Sensor 2 status to Uncalibrated.
jam3	**jam_sensor(s3)**	Forces Sensor 3 status to Uncalibrated.

IndiGolog – Reactive Controller

Exogenous Action: Jam1

The robot does not react immediately, but as soon as the conditions require it (activation of C), it detects the fault and repairs it.

Injection:

```
EM(3): Exogenous action occurred: [jam_sensor(s1),device(simulator)]
INDI(0): Received exogenous actions: [jam_sensor(s1)]
```

Reaction of the controller:

```
INFO(2): Sending action for execution: calibrate(s1)
EM(2): Send action to execute: [11,calibrate(s1),simulator,calibrate(s1)]
ACTION: Action EXECUTED: [[calibrate(s1),11],sensing(ok)]
```

IndiGolog – Legality Task

Legality Success

Plan tested: *open_container -> take_key -> open_panel*

```
INFO(2): Sending action: open_container
ACTION: Action EXECUTED: [[open_container,2],sensing(ok)]

INFO(2): Sending action: take_key
ACTION: Action EXECUTED: [[take_key,3],sensing(ok)]

INFO(2): Sending action: open_panel
ACTION: Action EXECUTED: [[open_panel,4],sensing(ok)]

INFO(2): Sending action: say(Sequence finished correctly (LEGAL).)
ACTION: Action EXECUTED: [[say(Sequence finished correctly (LEGAL).),5],sensing(o

PROGRAM: Program has executed to completion!!
```

Legality Fail

Plan tested: *open_container -> open_panel -> take_key*

```
ACTION: Action EXECUTED: [[open_container,2],sensing(ok)]

INFO(2): Waiting step for 1 seconds...

PROGRAM: Program fails:
         [[],open_panel,say(ERROR: This should not be reached.)]
 ...at history:
          [open_container,say(Testing ILLEGAL sequence (Expect Fail)...)]
```

IndiGolog – Projection Task

Projection Task Success:

Plan Tested: *open_container (0), take_key (0), open_panel (0), activate_auto(a) (15)*

```
INFO(2): Sending action: activate_auto(a)
ACTION: Action EXECUTED: [[activate_auto(a),5],sensing(ok)]

INFO(2): Sending action: say(Projection Result: TRUE (Battery check passed).)
ACTION: Action EXECUTED: [[say(Projection Result: TRUE...),6],sensing(ok)]

PROGRAM: Program has executed to completion!!
```

Projection Task Fail:

Plan Tested: *open_container (0), take_key (0), open_panel (0), activate_auto(a) (15)*

```
PROGRAM: Program fails:
         [[],?(battery>195),say(ERROR: This should not be reached.)]
 ...at history:
          [activate_auto(a),open_panel,take_key,...]
```