# Mars Exploration using PDDL and IndiGolog


SAPIENZA
UNIVERSITÀ DI ROMA

Ettore Branca 1965733
Francesco Caracci 1915689

# Project Idea

The idea for our project is an exploration task in a partially known environment, where autonomous agents, such as rovers and drones, collaborate to scan and explore areas and to collect resources.

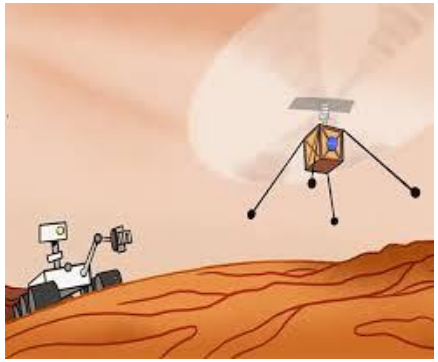
# Environment Description

The environment consists of a base station, multiple unknown locations and various obstacles, such as rocks and pits.

- **Drones** can fly over locations, scan areas for obstacles and determine traversability before rovers proceed.
- **Rovers** can move between traversable locations, explore areas and collect resources.
- The mission involves resource collection while minimizing energy consumption.

# PDDL: domain description

```
(:types
    agent location obstacle resource - object
    rover drone - agent
    base unknown_area - location
)

(:predicates
    (at ?agent - agent ?loc - location)
    (adjacent ?from ?to - location) 2👁
    (scanned ?loc - location) 2👁 1☑
    (traversable ?loc - location) 2👁 1☒
    (obstacle_at ?obst - obstacle ?loc - location) 2👁
    (explored ?loc - location) 2👁 1☑
    (resource_at ?res - resource ?loc - location) 1👁 1☒
    (collected ?res) 1☑
    (charging_station ?loc - location) 1👁
)

(:functions (energy-level ?agent - agent) 3👁 3↘ 1≔
            (total-cost) 6↗
)
```

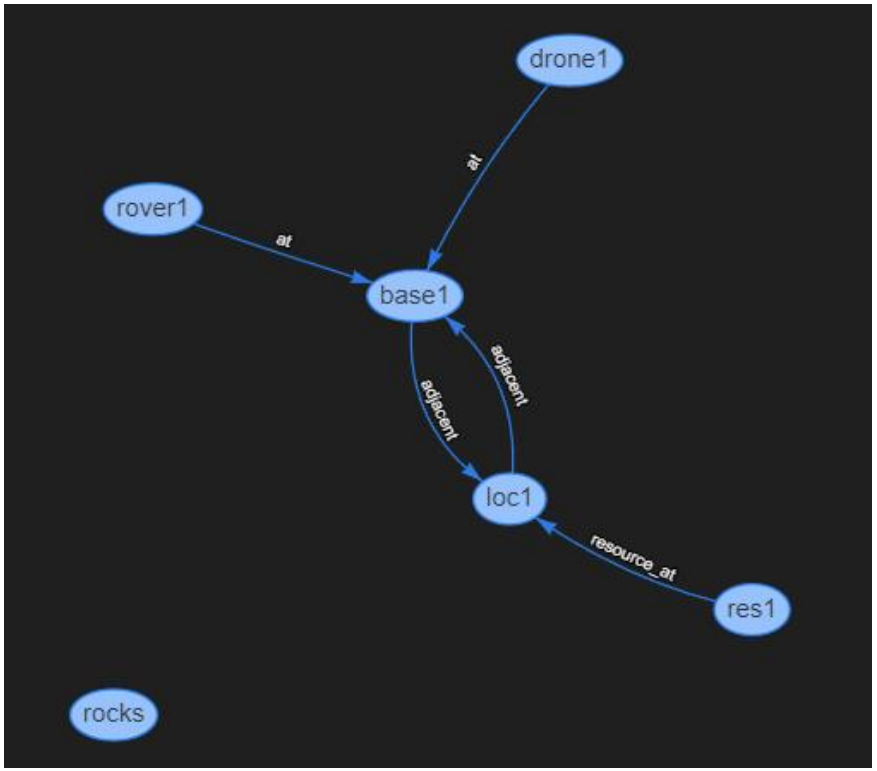


# PDDL: domain description

| Action | Description | Prerequisites | Effects |
|---|---|---|---|
| **Fly** | The drone moves between two adjacent locations. | - The drone is in the curret location;<br>- The destination is adjacent to the current location;<br>- The drone has sufficient energy. | - The drone changes its position to the destination;<br>- The energy decreases by 1. |
| **Scan** | The drone scans an unknown location. | The drone is at the location to be scanned. | The location is marked as scanned and, if no obstacles are present, as traversable. |
| **Move** | The rover travels between two traversable locations. | - The rover is in the current location;<br>- The destination location is adjacent, scanned and traversable;<br>- The rover has sufficient energy. | - The rover moves to the destination;<br>- The energy decreases by 1 unit. |
| **Explore** | The rover explores a location to gather information. | - The rover is at the current location;<br>- The location has not been explored yet;<br>- The rover has sufficient energy. | - The location is marked as explored;<br>- The energy decreases by 2. |
| **Collect** | The rover collects resources from an explored location. | - The rover is at the current location;<br>- The location is explored;<br>- The resource is present at the location. | The resource is collected and no longer available. |
| **Recharge** | The agent recharges its energy at the charging station. | The agent is in a location with a charging station. | The energy level is restored to 10 units. |

# PDDL: problem 1 description

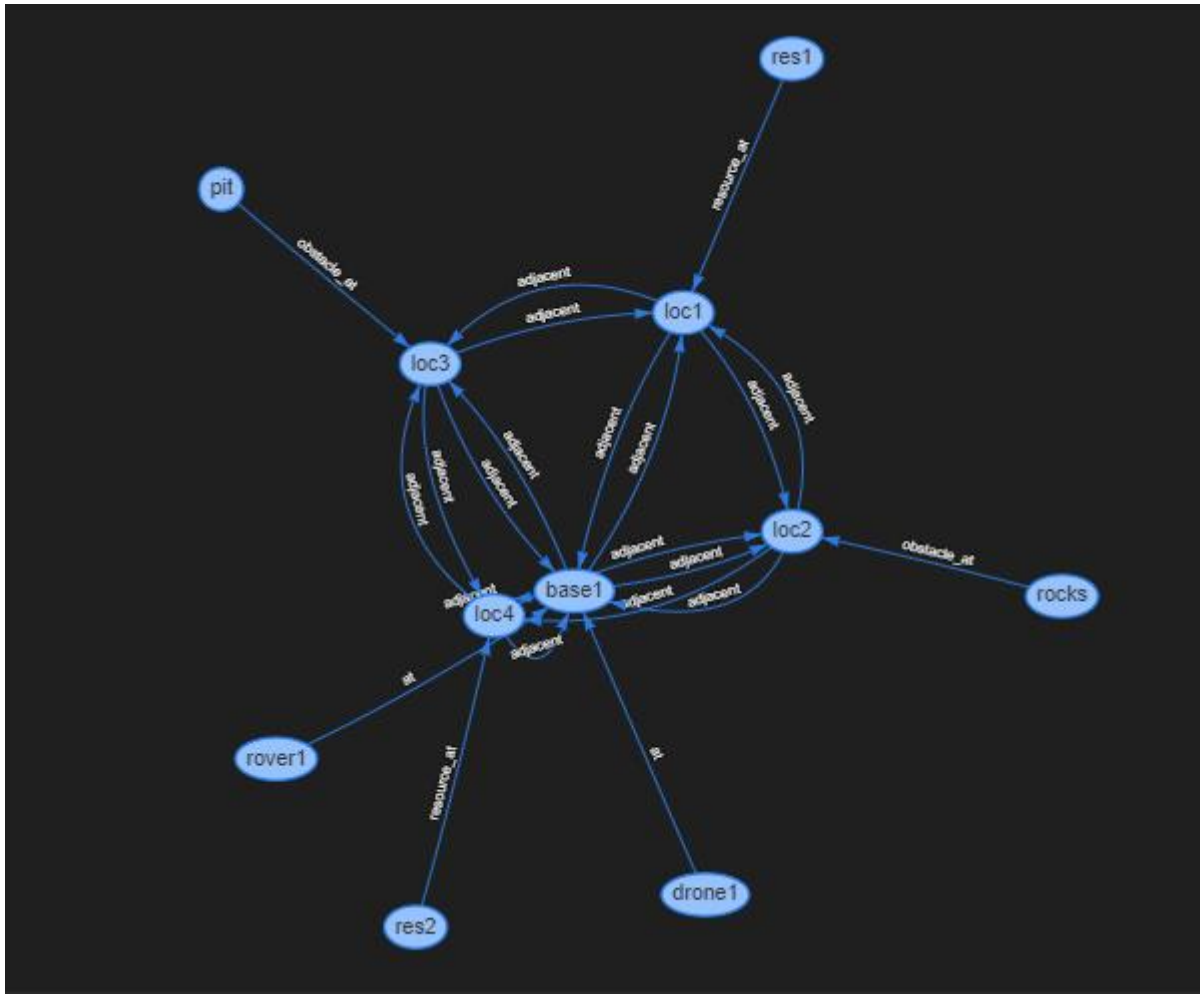
```
(:init
    ;; Initial positions
    (at rover1 base1)
    (at drone1 base1)
    ;; Assume base is known and safe
    (scanned base1)
    (traversable base1)
    (explored base1)
    ;; Adjacency relationships (assume bidirectional travel)
    (adjacent base1 loc1)
    (adjacent loc1 base1)
    ;; Charging station available at the base
    (charging_station base1)
    ;; Energy levels
    (= (energy-level rover1) 0)
    (= (energy-level drone1) 0)
    ;; Resource location
    (resource_at res1 loc1)
    ;; Total cost counter initialization
    (= (total-cost) 0)
)
(:goal (and
            (collected res1)
            (at rover1 base1)
            (at drone1 base1)
))
(:metric minimize (total-cost))
)
```



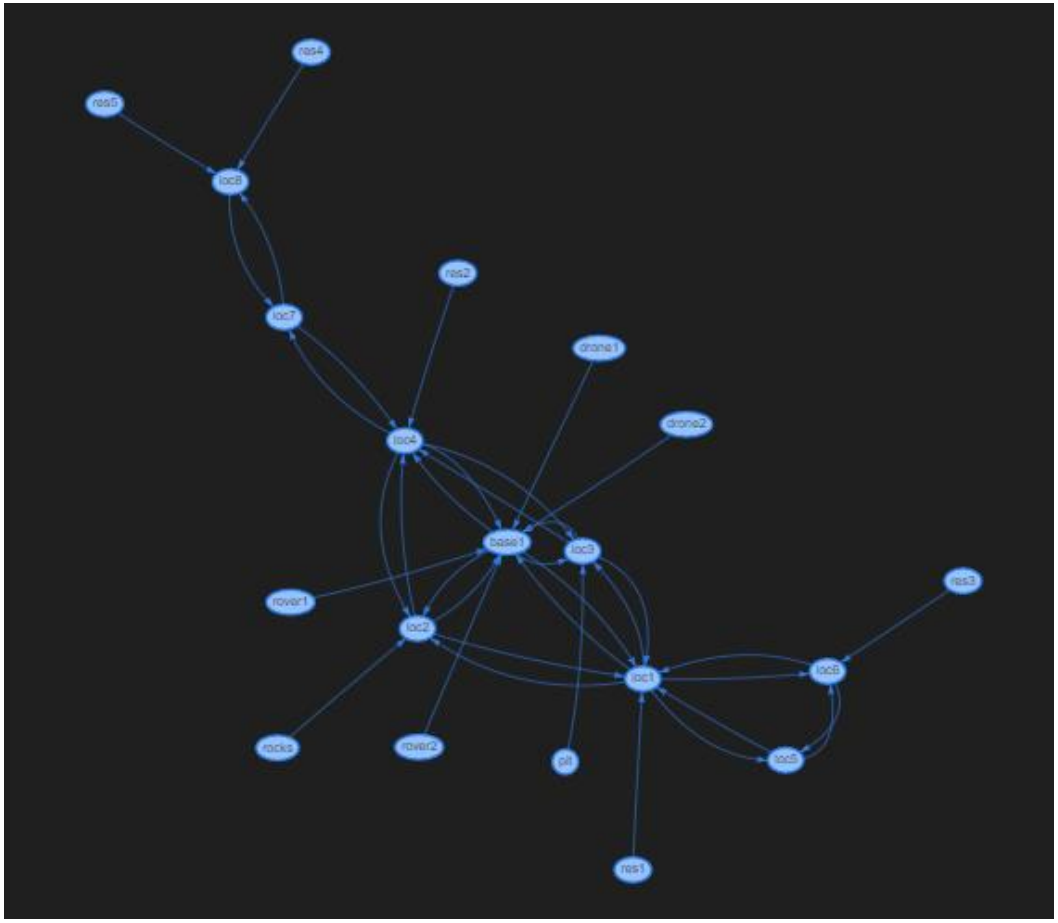
# PDDL: problem 2 description

```
(:objects
  rover1 - rover
  drone1 - drone
  base1 - base
  loc1 loc2 loc3 loc4 - unknown_area
  res1 res2 - resource
  rocks pit - obstacle
```

```
  (:goal (and
          (collected res1)
          (collected res2)
          (at rover1 base1)
          (at drone1 base1)
))
  (:metric minimize (total-cost))
```



# PDDL: problem 3 description




```
(:objects
  rover1 rover2 - rover
  drone1 drone2 - drone
  base1 - base
  loc1 loc2 loc3 loc4 loc5 loc6 loc7 loc8 - unknown_area
  res1 res2 res3 res4 res5 - resource
  rocks pit - obstacle
```

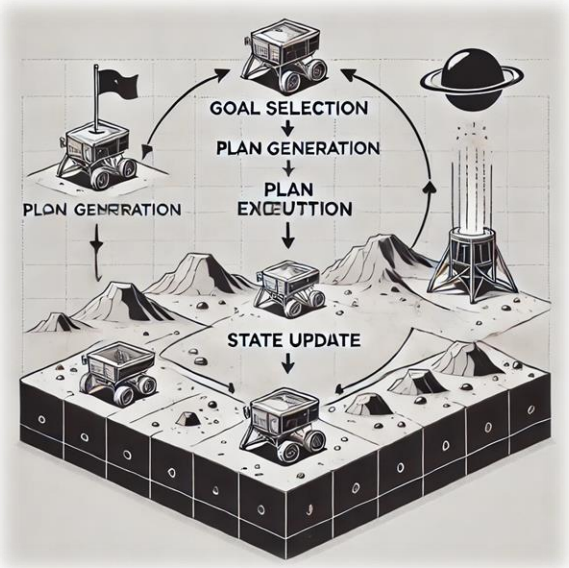
```
(:goal (and
        (collected res5)
        (collected res4)
        (collected res3)
        (collected res2)
        (collected res1)
        (at rover1 base1)
        (at drone1 base1)
        (at rover2 base1)
        (at drone2 base1)

(:metric minimize (total-cost))
```

# Solver and Heuristics

The planning component has been handled using **ENHSP** (Expressive Numeric Heuristic Search Planner), a PDDL-based planner that supports Classical and Numeric Planning (PDDL2.1).

To optimize planning, we have evaluated the following heuristics:

- **Sat-hadd**: *Greedy Best First Search* with numeric *hadd.*
- **Opt-hmax**: *A** with numeric *hmax.*



# PDDL: problem 1 solution

```
Running Greedy Best First Search
h(n = s_0)=48.0
 g(n)= 5.0 h(n)=28.0
 g(n)= 10.0 h(n)=13.0
 g(n)= 11.0 h(n)=10.0
 g(n)= 12.0 h(n)=6.0
 g(n)= 13.0 h(n)=5.0
 g(n)= 14.0 h(n)=4.0
 g(n)= 16.0 h(n)=2.0
 g(n)= 17.0 h(n)=1.0
 g(n)= 18.0 h(n)=0.0
Problem Solved

Found Plan:
0.0: (recharge drone1 base1)
1.0: (recharge rover1 base1)
2.0: (fly drone1 base1 loc1)
3.0: (scan drone1 loc1)
4.0: (fly drone1 loc1 base1)
5.0: (move rover1 base1 loc1)
6.0: (explore rover1 loc1)
7.0: (collect rover1 res1 loc1)
8.0: (move rover1 loc1 base1)

Plan-Length:9
Metric (Search):18.0
Planning Time (msec): 382
Heuristic Time (msec): 2
Search Time (msec): 18
Expanded Nodes:10
States Evaluated:23
```

```
Running WA-STAR
h(n = s_0)=6.0
f(n) = 6.0 (Expanded Nodes: 0, E
f(n) = 11.0 (Expanded Nodes: 1,
f(n) = 12.0 (Expanded Nodes: 5,
f(n) = 13.0 (Expanded Nodes: 6,
f(n) = 14.0 (Expanded Nodes: 9,
f(n) = 15.0 (Expanded Nodes: 10,
f(n) = 16.0 (Expanded Nodes: 13,
f(n) = 17.0 (Expanded Nodes: 19,
f(n) = 18.0 (Expanded Nodes: 27,
Problem Solved

Found Plan:
0.0: (recharge drone1 base1)
1.0: (fly drone1 base1 loc1)
2.0: (scan drone1 loc1)
3.0: (recharge rover1 base1)
4.0: (move rover1 base1 loc1)
5.0: (explore rover1 loc1)
6.0: (collect rover1 res1 loc1)
7.0: (move rover1 loc1 base1)
8.0: (fly drone1 loc1 base1)

Plan-Length:9
Metric (Search):18.0
Planning Time (msec): 298
Heuristic Time (msec): 15
Search Time (msec): 31
Expanded Nodes:29
States Evaluated:53
```

# PDDL: problem 2 solution

```
Running Greedy Best First Search
h(n = s_0)=96.0
 g(n)= 5.0 h(n)=56.0
 g(n)= 10.0 h(n)=26.0
 g(n)= 12.0 h(n)=23.0
 g(n)= 13.0 h(n)=18.0
 g(n)= 14.0 h(n)=15.0
 g(n)= 15.0 h(n)=11.0
 g(n)= 16.0 h(n)=10.0
 g(n)= 19.0 h(n)=9.0
 g(n)= 20.0 h(n)=7.0
 g(n)= 23.0 h(n)=5.0
 g(n)= 24.0 h(n)=4.0
 g(n)= 25.0 h(n)=2.0
 g(n)= 27.0 h(n)=1.0
 g(n)= 28.0 h(n)=0.0
Problem Solved

Found Plan:
0.0: (recharge drone1 base1)
1.0: (recharge rover1 base1)
2.0: (fly drone1 base1 loc1)
3.0: (scan drone1 loc1)
4.0: (fly drone1 loc1 base1)
5.0: (fly drone1 base1 loc4)
6.0: (scan drone1 loc4)
7.0: (fly drone1 loc4 base1)
8.0: (move rover1 base1 loc1)
9.0: (explore rover1 loc1)
10.0: (move rover1 loc1 base1)
11.0: (move rover1 base1 loc4)
12.0: (explore rover1 loc4)
13.0: (collect rover1 res2 loc4)
14.0: (move rover1 loc4 base1)
15.0: (move rover1 base1 loc1)
16.0: (collect rover1 res1 loc1)
17.0: (move rover1 loc1 base1)

Plan-Length:18
Metric (Search):28.0
Planning Time (msec): 340
Heuristic Time (msec): 10
Search Time (msec): 33
Expanded Nodes:30
States Evaluated:150
```

```
Running WA-STAR
h(n = s_0)=6.0
f(n) = 6.0 (Expanded Nodes: 0, Ev
f(n) = 11.0 (Expanded Nodes: 1, E
f(n) = 12.0 (Expanded Nodes: 3, E
f(n) = 13.0 (Expanded Nodes: 5, E
f(n) = 14.0 (Expanded Nodes: 12,
f(n) = 15.0 (Expanded Nodes: 33,
f(n) = 16.0 (Expanded Nodes: 75,
f(n) = 17.0 (Expanded Nodes: 138,
f(n) = 18.0 (Expanded Nodes: 213,
f(n) = 19.0 (Expanded Nodes: 298,
f(n) = 20.0 (Expanded Nodes: 407,
f(n) = 21.0 (Expanded Nodes: 553,
f(n) = 22.0 (Expanded Nodes: 761,
f(n) = 23.0 (Expanded Nodes: 1011
f(n) = 24.0 (Expanded Nodes: 1336
f(n) = 25.0 (Expanded Nodes: 1813
f(n) = 26.0 (Expanded Nodes: 2534
Problem Solved

Found Plan:
0.0: (recharge drone1 base1)
1.0: (fly drone1 base1 loc1)
2.0: (scan drone1 loc1)
3.0: (fly drone1 loc1 loc2)
4.0: (fly drone1 loc2 loc4)
5.0: (recharge rover1 base1)
6.0: (move rover1 base1 loc1)
7.0: (explore rover1 loc1)
8.0: (collect rover1 res1 loc1)
9.0: (scan drone1 loc4)
10.0: (move rover1 loc1 base1)
11.0: (move rover1 base1 loc4)
12.0: (explore rover1 loc4)
13.0: (collect rover1 res2 loc4)
14.0: (move rover1 loc4 base1)
15.0: (fly drone1 loc4 base1)

Plan-Length:16
Metric (Search):26.0
Planning Time (msec): 460
Heuristic Time (msec): 89
Search Time (msec): 154
Expanded Nodes:2538
States Evaluated:4863
```

# PDDL: problem 3 solution

```
Running Greedy Best First Search
h(n = s_0)=608.0
```

```
 g(n)= 56.0 h(n)=2.0
 g(n)= 57.0 h(n)=1.0
 g(n)= 58.0 h(n)=0.0
Problem Solved

Found Plan:
0.0: (recharge drone2 base1)
1.0: (recharge rover2 base1)
2.0: (fly drone2 base1 loc4)
3.0: (scan drone2 loc4)
4.0: (recharge drone1 base1)
5.0: (fly drone2 loc4 loc7)
6.0: (fly drone1 base1 loc1)
7.0: (scan drone1 loc1)
8.0: (scan drone2 loc7)
9.0: (fly drone2 loc7 loc8)
10.0: (scan drone2 loc8)
11.0: (fly drone1 loc1 loc6)
12.0: (scan drone1 loc6)
13.0: (move rover2 base1 loc4)
14.0: (recharge rover1 base1)
15.0: (move rover1 base1 loc1)
16.0: (explore rover2 loc4)
17.0: (move rover2 loc4 loc7)
18.0: (move rover2 loc7 loc8)
19.0: (explore rover2 loc8)
20.0: (explore rover1 loc1)
21.0: (fly drone1 loc6 loc1)
22.0: (fly drone1 loc1 base1)
23.0: (fly drone2 loc8 loc7)
24.0: (fly drone2 loc7 loc4)
25.0: (fly drone2 loc4 base1)
26.0: (collect rover1 res1 loc1)
27.0: (move rover1 loc1 loc6)
28.0: (explore rover1 loc6)
29.0: (collect rover1 res3 loc6)
30.0: (move rover1 loc6 loc1)
31.0: (move rover1 loc1 base1)
32.0: (collect rover2 res5 loc8)
33.0: (collect rover2 res4 loc8)
34.0: (move rover2 loc8 loc7)
35.0: (move rover2 loc7 loc4)
36.0: (collect rover2 res2 loc4)
37.0: (move rover2 loc4 base1)

Plan-Length:38
Metric (Search):58.0
Planning Time (msec): 470
Heuristic Time (msec): 50
Search Time (msec): 87
Expanded Nodes:39
States Evaluated:416
```

```
Running WA-STAR
h(n = s_0)=8.0
```

```
Running WA-STAR
h(n = s_0)=8.0
f(n) = 8.0 (Expanded Nodes: 0, Evaluated States: 0, Time: 0.015) Frontier Size: 0
f(n) = 13.0 (Expanded Nodes: 1, Evaluated States: 4, Time: 0.031) Frontier Size: 3
f(n) = 14.0 (Expanded Nodes: 7, Evaluated States: 34, Time: 0.031) Frontier Size: 27
f(n) = 15.0 (Expanded Nodes: 17, Evaluated States: 96, Time: 0.057) Frontier Size: 79
f(n) = 16.0 (Expanded Nodes: 47, Evaluated States: 272, Time: 0.083) Frontier Size: 225
f(n) = 17.0 (Expanded Nodes: 121, Evaluated States: 662, Time: 0.11) Frontier Size: 541
f(n) = 18.0 (Expanded Nodes: 257, Evaluated States: 1342, Time: 0.141) Frontier Size: 1
f(n) = 19.0 (Expanded Nodes: 525, Evaluated States: 2605, Time: 0.205) Frontier Size: 2
f(n) = 20.0 (Expanded Nodes: 983, Evaluated States: 4731, Time: 0.283) Frontier Size: 3
f(n) = 21.0 (Expanded Nodes: 1768, Evaluated States: 8248, Time: 0.409) Frontier Size: 6
f(n) = 22.0 (Expanded Nodes: 3148, Evaluated States: 14225, Time: 0.629) Frontier Size:
f(n) = 23.0 (Expanded Nodes: 5557, Evaluated States: 24540, Time: 0.991) Frontier Size:
f(n) = 24.0 (Expanded Nodes: 9814, Evaluated States: 42302, Time: 1.604) Frontier Size:
f(n) = 25.0 (Expanded Nodes: 17072, Evaluated States: 72010, Time: 2.609) Frontier Size:
f(n) = 26.0 (Expanded Nodes: 29143, Evaluated States: 120349, Time: 4.237) Frontier Size
f(n) = 27.0 (Expanded Nodes: 49007, Evaluated States: 195928, Time: 6.77) Frontier Size:
-------------Time: 10s ; Expanded Nodes: 72851 (Avg-Speed 7285.0 n/s); Evaluated States:
f(n) = 28.0 (Expanded Nodes: 81542, Evaluated States: 313547, Time: 10.788) Frontier Siz
f(n) = 29.0 (Expanded Nodes: 134821, Evaluated States: 500247, Time: 17.387) Frontier Si
-------------Time: 20s ; Expanded Nodes: 148664 (Avg-Speed 7433.0 n/s); Evaluated States
```

```
-Time: 489s ; Expanded Nodes: 1605955 (Avg-Speed 3284.0 n/s)
 thread "main" java.lang.OutOfMemoryError: Java heap space:
```

# IndiGolog

Considering that our environment is dynamic and partially observable, traditional offline planning approaches, like PDDL, are not always sufficient. This is why IndiGolog becomes essential.
It is a high-level programming language that provides an imperative approach, allowing for real-time execution, **exogenous events** and dynamic changes handling and **agent-sensor interaction**.
In this domain, the mission can be more autonomous and resilient to unexpected challenges.

# IndiGolog: Fluents

```
rel_fluent(at(_,_)).
rel_fluent(scanned(_)).
rel_fluent(traversable(_)).
rel_fluent(explored(_)).
rel_fluent(resource_at(_,_)).
rel_fluent(collected(_)).
rel_fluent(adjacent(_,_)).

rel_fluent(sandstorm_at(_)).

fun_fluent(energy_level(_)).

proc(enough_energy(Agent), energy_level(Agent) > 0).
```

# IndiGolog: Primitive Actions

```
%Fly action
prim_action(fly(Drone, From, To)):-
    drone(Drone), location(From), area(To).
poss(fly(Drone, From, To),
    and(at(Drone, From), and(adjacent(From, To), enough_energy(Drone)))).

% Scan action
prim_action(scan(Drone, Area)):-
    drone(Drone), area(Area).
poss(scan(Drone, Area), and(at(Drone, Area), enough_energy(Drone))).

% Move action
prim_action(move(Rover, From, To)):-
    rover(Rover), location(From), location(To).
poss(move(Rover, From, To),
        and(at(Rover, From), and(adjacent(From, To), and(traversable(To), enough_energy(Rover))))).

% Explore action
prim_action(explore(Rover, Area)):-
    rover(Rover), area(Area).
poss(explore(Rover, Area), and(at(Rover, Area), and(traversable(Area), enough_energy(Rover)))).

% Collect action
prim_action(collect(Rover, Resource, Location)):-
    rover(Rover), resource(Resource), area(Location).
poss(collect(Rover, Resource, Location),
    and(at(Rover, Location), and(resource_at(Resource, Location),
    and(neg(collected(Resource)), and(explored(Location), enough_energy(Rover)))))).

% Handle Sandstorm action
prim_action(handle_sandstorm(Area)) :- area(Area).
poss(handle_sandstorm(Area), sandstorm_at(Area)).
```

# IndiGolog: Successor State Axioms

```
causes_false(fly(Drone,From,_To), at(Drone,From), true).
causes_true(fly(Drone,_From,To), at(Drone, To), true).
causes_val(fly(Drone,_From,_To), energy_level(Drone), E, E is energy_level(Drone)-1).

causes_true(scan(_Drone, Area), scanned(Area), true).
causes_val(scan(Drone, Area), energy_level(Drone), E, E is energy_level(Drone)-1).

causes_true(move(Rover,_From,To), at(Rover, To), true).
causes_false(move(Rover, From, _To), at(Rover, From), true).
causes_val(move(Rover,_From,_To), energy_level(Rover), E, E is energy_level(Rover)-1).

causes_true(explore(_Rover, Area), explored(Area), true).
causes_val(explore(Drone, Area), energy_level(Drone), E, E is energy_level(Drone)-1).

causes_true(collect(_Rover, Resource, _Location), collected(Resource), true).
causes_false(collect(_Rover, Resource, Location), resource_at(Resource, Location), true).
causes_val(collect(Rover, _Resource, _Location), energy_level(Rover), E, E is energy_level(Rover)-1).

causes_false(handle_sandstorm(Area), sandstorm_at(Area), true).
```

# IndiGolog: Sensing and Exogenous actions

```
senses(detect_no_obstacle(Drone, Area), traversable(Area)).
prim_action(detect_no_obstacle(Drone, Area)):- drone(Drone), area(Area).
poss(detect_no_obstacle(Drone, Area), and(at(Drone, Area), scanned(Area))).
```

```
exog_action(sandstorm(Area)):- area(Area).
causes_true(sandstorm(Area), sandstorm_at(Area), true).
causes_false(sandstorm(Area), scanned(Area), true).
causes_false(sandstorm(Area), traversable(Area), true).
```

# IndiGolog: Procedures

```
proc(drone_scan_area(Drone, To),
        [fly(Drone, From, To), scan(Drone, To), detect_no_obstacle(Drone, To)]).

proc(safe_move(Rover, To),
           [drone_scan_area(Drone, To),
           if(traversable(To),
              move(Rover,From, To), [])]).

proc(explore_and_collect(Location, Resource),
     [safe_move(Rover, Location),
      if(neg(explored(Location)), explore(Rover, Location), []),
      if(resource_at(Resource, Location), collect(Rover, Resource, Location), [])]).
```

# IndiGolog: Tasks

```
%Controller to collect a generic resource
proc(control(collect_resource),
       if(some(r, neg(collected(r))),
          pi([l,r], explore_and_collect(l,r)),
          [])).

% Controller to collect all resources
proc(control(collect_all),
    while(some([r,l], (and(resource_at(r,l), neg(collected(r))))),
       [pi([l,r], explore_and_collect(l, r))])).

% Controller to collect all resources and handle exogenous events
proc(control(collect_all_exo),
     [prioritized_interrupts([
        interrupt(some(l, sandstorm_at(l)), pi(l, [handle_sandstorm(l), scan(Drone, l), detect_no_obstacle(Drone, l)])),
        interrupt(some([r,l], (and(resource_at(r,l), neg(collected(r))))), pi([l,r],explore_and_collect(l,r))),
        interrupt(true, ?(wait_exog_action))
        ])]).
```

# IndiGolog: Task 1 solution

```
PROGRAM: Program has executed to completion!! History done:
         [collect(rover1,res1,loc1),explore(rover1,loc1),move(rover1,base1,loc1),e(detect_no_obstacle(drone1,loc1),true),
detect_no_obstacle(drone1,loc1),scan(drone1,loc1),fly(drone1,base1,loc1)]
END: Execution finished. Closing modules...
END: INDIGOLOG is finishing...
END: Finalizing PROJECTOR...
END: PROJECTOR was finalized successfully.
END: Finalizing ENVIRONMENT MANAGER...
EM(1): EM completed with 6 executed actions
END: ENVIRONMENT MANAGER was finalized successfully.
END: Everything finished - HALTING TOP-LEVEL CONTROLLER
true.
```

# IndiGolog: Task 2 solution

```
PROGRAM: Program has executed to completion!! History done:
         [collect(rover1,res2,loc2),explore(rover1,loc2),move(rover1,loc1,loc2),e(detect_no_obstacle(drone1,loc2),true)
,detect_no_obstacle(drone1,loc2),scan(drone1,loc2),fly(drone1,loc1,loc2),collect(rover1,res1,loc1),explore(rover1,loc1)
,move(rover1,base1,loc1),e(detect_no_obstacle(drone1,loc1),true),detect_no_obstacle(drone1,loc1),scan(drone1,loc1),fly(
drone1,base1,loc1)]
END: Execution finished. Closing modules...
END: INDIGOLOG is finishing...
END: Finalizing PROJECTOR...
END: PROJECTOR was finalized successfully.
END: Finalizing ENVIRONMENT MANAGER...
EM(1): EM completed with 12 executed actions
END: ENVIRONMENT MANAGER was finalized successfully.
END: Everything finished - HALTING TOP-LEVEL CONTROLLER
true.
```

# IndiGolog: Task 3 solution

```
PROGRAM: Program has executed to completion!! History done:
        [collect(rover1,res2,loc2),explore(rover1,loc2),move(rover1,loc1,loc2),e(detect_no_obstacle(drone1,loc2),true),detect_no_obstacle(drone1,loc
2),scan(drone1,loc2),handle_sandstorm(loc2),sandstorm(loc2),e(detect_no_obstacle(drone1,loc2),true),detect_no_obstacle(drone1,loc2),scan(drone1,loc2)
,fly(drone1,loc1,loc2),collect(rover1,res1,loc1),explore(rover1,loc1),move(rover1,base1,loc1),e(detect_no_obstacle(drone1,loc1),true),detect_no_obsta
cle(drone1,loc1),scan(drone1,loc1),fly(drone1,base1,loc1)]
END: Execution finished. Closing modules...
END: INDIGOLOG is finishing...
END: Finalizing PROJECTOR...
END: PROJECTOR was finalized successfully.
END: Finalizing ENVIRONMENT MANAGER...
EM(1): EM completed with 15 executed actions
END: ENVIRONMENT MANAGER was finalized successfully.
END: Everything finished - HALTING TOP-LEVEL CONTROLLER
true.
```

# Thanks for the attention!