Automated (AI) Planning - Hands-on

Autonomous Software Agents - Lab

> Marco Robol - marco.robol@unitn.it

Preperation

- You can work in teams or alone
- On your laptop get: http://www.cs.rochester.edu/u/kautz/satplan/blackbox/
- Use a text editor to edit pddl files
- Use the console to launch Blackbox:
  ```
  blackbox -o domain.pddl -f prob004-log-a.pddl
  ```

Alternatively use an online planner:

- https://web-planner.herokuapp.com/
- http://planning.domains/

Blocks world

```
;; problem file: blocksworld-prob1.pddl

(define (problem blocksworld-prob1)
  (:domain blocksworld)
  (:objects a b)
  (:init (on-table a) (on-table b) (clear a) (clear b))
  (:goal (and (on a b))))
```

To Do

1. Implement the domain file for blocksworld (See partial domain file in next slide)
2. Define more complex problems for the blocksworld domain
3. Redefine the domain but with only 2 actions. And what about the problem?

The blocks world domain file (partial)

```
;; domain file: blocksworld-domain.pddl
(define (domain blocksworld)
  (:requirements :strips)

  (:predicates (clear ?x)
               (on-table ?x)
               (holding ?x)
               (on ?x ?y))

  (:action pickup
           :parameters (?ob)
           :precondition (and (clear ?ob) (on-table ?ob))
           :effect (and (holding ?ob) (not (clear ?ob))
                        (not (on-table ?ob))))
```

Blocks world - Solution 1/2

…

Blocks world - Solution 2/2

...

Tower of Hanoi

- Only one disk can be moved at a time.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack i.e. a disk can only be moved if it is the uppermost disk on a stack.
- No disk may be placed on top of a smaller disk.
- Problem: 3 discs to be moved from peg1 to peg2.

Tower of Hanoi - Domain - Solution

...

Tower of Hanoi - Problem (3 discs) - Solution

...

Rabbit

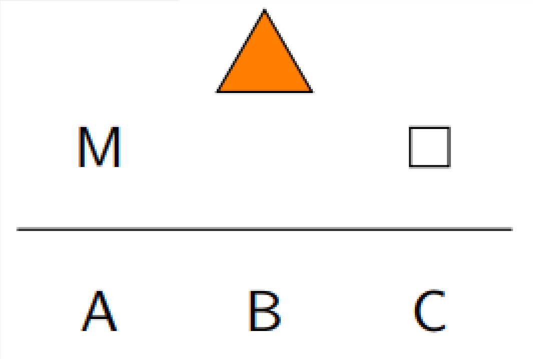
A **rabbit** want to get a bunch of **carrots** that are tied in the **ceiling**. The rabbit cannot reach the carrots by itself, but there is a **chair** on the **floor** that the rabbit can use to reach the carrots.

Rabbit

- There is a **rabbit** in a room with a **box** and **carrots** hanging from the ceiling.
- The rabbit and the box has a low height. The carrots have a high height.
  If the rabbit climbs on the box, its height becomes high.
- The rabbit can
  - **Go** from one place to another,
  - **Push** an object from one place to another,
  - **ClimbUp** onto or **ClimbDown** from an object,
  - **Grasp** or **Ungrasp** an object (if the rabbit and object are in the same place at the same height.)

Rabbit: To Do

- Create the domain file. What are the predicates? What are the actions?
- Create a problem file:
    - Initially, the rabbit is at A, the carrots at B, and the box at C.



    - The rabbit **wants** to be on the floor with the carrots.

Rabbit Problem: Clues

Predicates:

- Where are the objects vertically? (Height ?object ?pos)
- Where are the objects horizontally? (At ?object ?h)
- What is high? What is low? (High ?h) (Low ?l)
- What are the objects? Rabbit, chair, place, carrot

Rockets Domain

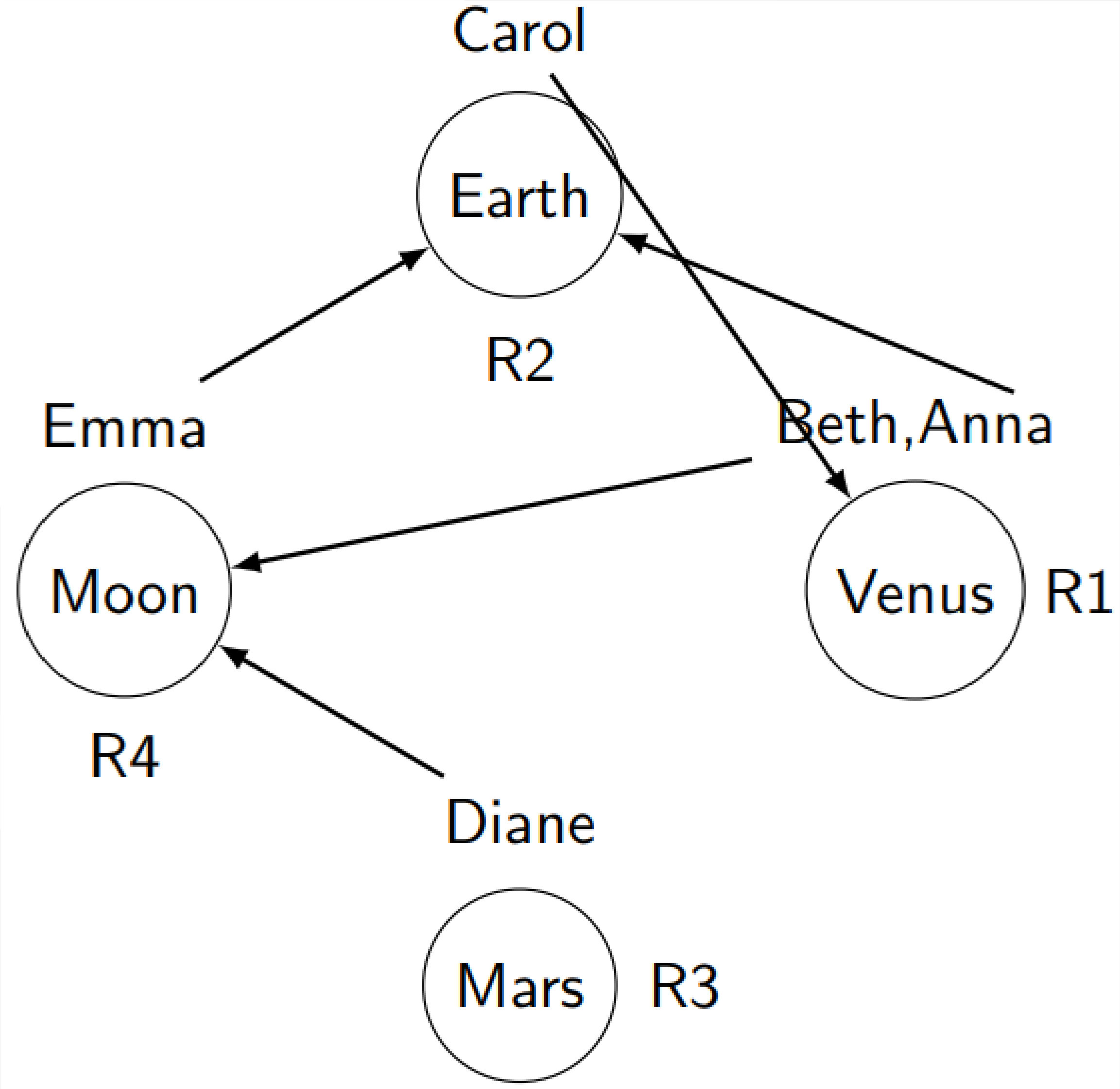
- There are **rockets** and **cargos** and **locations**.
- We want to transport **cargos** to the intended destionations.
- Each rocket fly only once, then it goes out of **fuel**.

Rockets - To Do

- Bob and Alice wants to go to Mars from Earth. There is one rocket available.
- Write a more complex problem on your own.

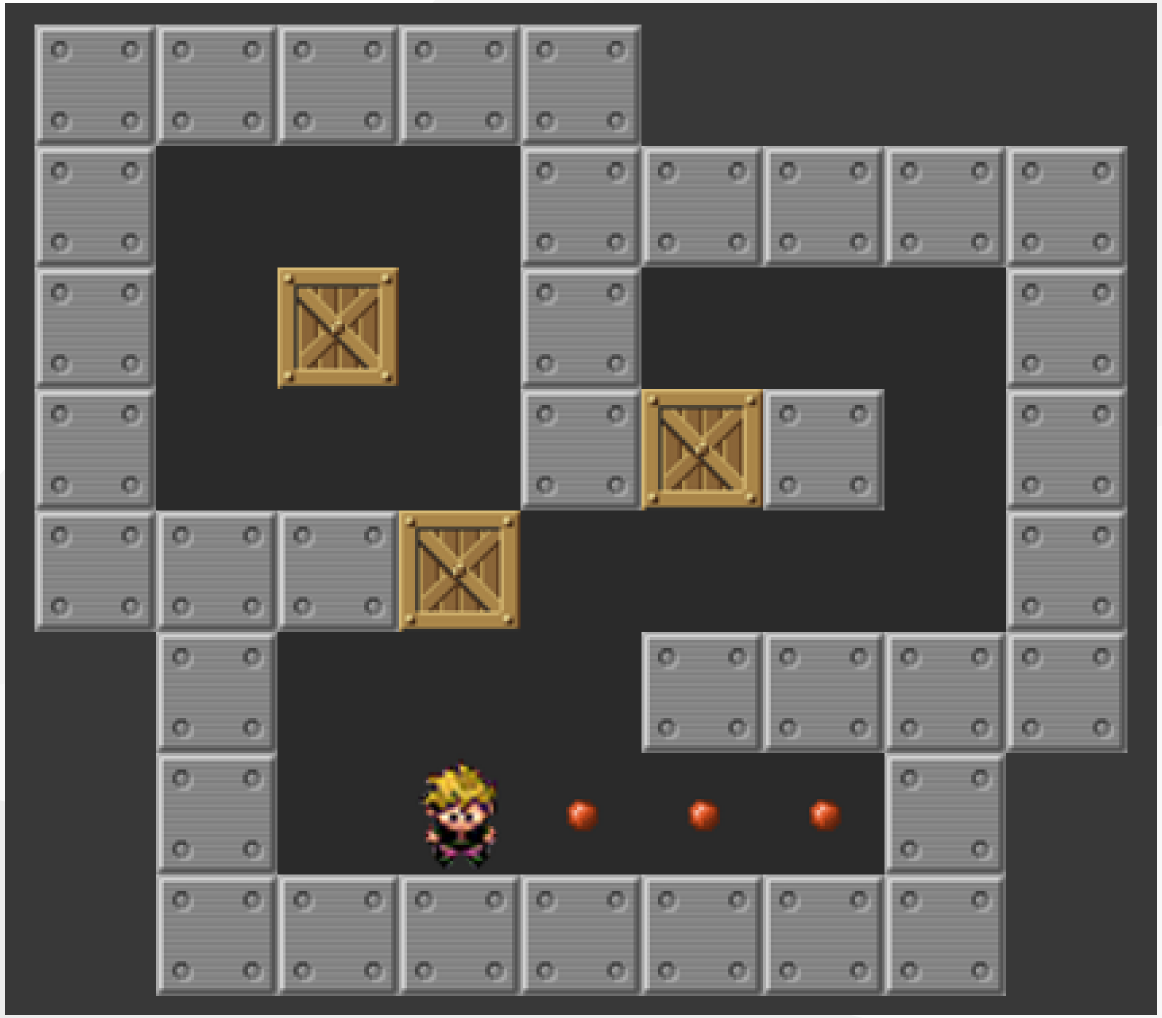
4 Rockets - To Do

Define this problem in pddl



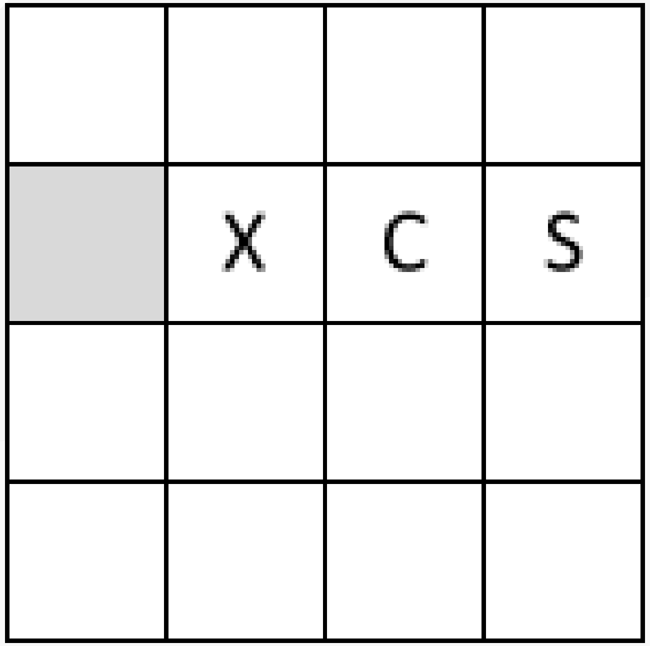
Sokoban

- Sokoban, which means "warehouse keeper", is a Japanese game created in 1982.
- You have to push crates to their proper location.
- You cannot pull the crates or step over them.
- You can only push one crate at a time, you cannot push a row of crates.
- Grid world representation.



Sokoban - To Do

- Create the domain file. What are the predicates? What are the actions?
- Create a problem file: (S: Sokoban, C: crate, X destination)



Sokoban - Clues

```
...
(:predicates (left ?x1 ?x2) (up ?y1 ?y2)
             (sokoban ?x ?y) (wall ?x ?y) (crate ?x ?y)
)
(:action move-left
  :parameters (?x1 ?x2 ?y)
...
(:action push-left
...
```

Conclusions

- We have seen **toy examples**.
- In real life planners are used to solve non-trivial problems in robotics, logistics, space misiion, and multiagent systems
- Agents perceive the world (initial state), and then act (using actions) to reach their goal state.

Questions?

marco.robol@unitn.it

Additional exercise 1

A robot postman is assigned to deliver daily mail in Trento. The robot loads the mail to its mail truck if the mail and the truck are at the same location, drives the truck to mail destination, and unloads the mail when it reaches to the destination. - Describe this domain in PDDL. Fill in the action definitions and add predicates if necessary.

```
;; STRIPS domain of the robo-postman
(define (domain robopost)
  (:requirements :strips)
  (:predicates (In ?mail ?truck) (At-truck ?truck ?location)
    (At-mail ?mail ?location) (Is-mail ?mail)
    (Is-truck ?truck) (Is-location ?location))
  (:action LOAD)
  (:action UNLOAD)
  (:action DRIVE)
)
```

Additional exercise 1

Write down the solution for the problem.

```
(define (problem robopostprob) (:domain robopost)
  (:objects pt m1 m2 trento povo rovereto)
  (:init (Is-truck pt) (Is-mail m1) (Is-mail m2)
         (Is-location trento) (Is-location povo)
         (Is-location rovereto) (At-truck pt rovereto)
         (At-mail m1 rovereto) (At-mail m2 trento)
  )
  (:goal (and (At-mail m1 trento) (At-mail m2 povo))
  )
)
```

Additional exercise 2

The cableway of Sardagna connects the small village with the city of Trento, with a ride of 4 minutes. The cableway is composed of two cable cars, when the first one is in Trento the second one is in Sardagna, or vice versa.

Describe this domain in PDDL. Proposed three actions are sufficient for this domain.

```
;; STRIPS domain of the cableway
(define (domain cableway)
  (:requirements :strips)
  (:predicates (Cable ?cable-station-A ?cable-station-B)
    (Cable-car-at ?cable-car ?cable-station)
    (Person-at ?person ?cable-station)
    (Person-on ?person ?cable-car) )
  (:action GET-ON )
  (:action GET-OFF )
  (:action RIDE )
)
```

Additional exercise 2

Write down a plan for the following problem

```
(define (problem cableway-pb1) (:domain cableway)
  (:objects carA carB luca matteo trento sardagna)
  (:init
    (Cable-car-at carA trento) (Cable-car-at carB sardagna)
    (Cable trento sardagna)
    (Person-at luca trento) (Person-at matteo sardagna)
  )
  (:goal (and
    (Person-at luca sardagna) (Person-at matteo trento) )
  )
)
```

For example:

1. GET-ON luca trento carA;
2. . . .