Amazon racking robots

Project report

Tobia Cargnello

Advanced topics in Artificial Intelligence 2 (650.045, 21W)

Contents

1 Introduction

This project is about finding reverse plans in a STRIPS domain. For this project a STRIPS domain has been implemented in PDDL and a simplified version of the same domain has been implemented in K language. Tools and encodings presented in the lectures have been used to find the reverse plans. Firtsly, the domain is presented along with the setup used to do this project. Then the implementation in PDDL and in K is presented is section 2. Some screenshots of the testing phase are shown in section 3 and finally the commands used in order to execute the tests are shown in section 4.

1.1 The domain

The domain idea comes from the project idea done for the Advanced Topics in Artificial Intelligence 1, which consisted in an implementation of the "value iteration" algorithm in Python and was not involving planning. In this domain, there are different robots in a warehouse, it can be imagined similar to an Amazon warehouse, that need to collect boxes in a goal location. In particular, the system considered is formed by robots that collect boxes into a goal position and robots that bring boxes to the start location if the dropping of the box to a goal location needs to be "undone". The boxes can be of normal size or they can be big boxes that are too heavy to be carried by just one robot. In that case, the robot needs to call for another robot, perform a composition with it in order to form a bigger robot capable of carrying the heavy boxes which can be then collected to the goal location. A robot can also open the boxes to inspect what is inside and also close the boxes. A robot is powered by a battery, therefore it needs to go to the charging station whenever its battery is low. In this domain there are some reversible actions which effect can be "undone". The goal in this project is to find the existing possible reverse plans.

1.2 Set up

For this project, the following tools have been used:

- clingo version 5.4.0;
- eclingo version 0.2.0;
- plasp version 3.1.1;
- qasp version 0.1.

In particular, clingo and eclingo have been set up in an anaconda environment with Python version 3.6.
All the test have been run on a machine with the following specifications:

- CPU: Intel Core i7-4720HQ CPU 2.60GHz × 8
- Memory(RAM): 16GB
- GPU: NVIDIA GeForce GTX 950M

- Operative System: EndeavourOS
- Editor: Visual Studio Code

2 Implementation

2.1 PDLL

There are 13 actions in total. The first action presented in *Listing* 1 allows a robot to go to the recharging station and get charged again. This action can be performed without any preconditions.

Listing 1: Go to the recharging station action

```
    (:action go_to_charge
        :effect (and
            (robot_charged)
        )
    )
```

In *Listing* 2 the action that allows to bring the box to the start location is presented. It can be imagined that this action is performed by a different kind of robot that picks the box and put it at the start location.

Listing 2: Bring a box to the start location

```
    (:action bring_box
        :effect (and
            (box_at_start)
        )
    )
```

In *Listing* 3 the action that allows a robot to go to the start position is presented. A robot can go to the start position only if it is charged.

Listing 3: Go to the box location action

```
    (:action go_to_start
        :precondition (and
            (robot_charged)
        )
        :effect (and
            (robot_at_start)
            (not (robot_at_goal))
        )
)
```

Once a robot is at the start location, it is charged and there is a box to pick up, it can pick it up as specified in the action presented in *Listing* 4.

Listing 4: Pick the box action

```
(:action pick_box
    :precondition (and
        (robot_at_start)
        (robot_charged)
        (box_at_start)
    )
    :effect (and
        (robot_carries_box)
    )
)
```

A robot can go to the goal position only if it carries a box to put there. To do that it has to be at the start location with a charged battery. The action is presented in *Listing* 5.

Listing 5: Go to goal location action

```
(:action go_to_goal
    :precondition (and
        (robot_charged)
        (box_at_start)
        (robot_carries_box)
    )
    :effect (and
        (not (box_at_start))
        (robot_at_goal)
        (not (robot_at_start))
    )
)
```

Once a robot has reached the goal position, it can drop the box as showed in *Listing* 6. This action causes the robot to be discharged.

Listing 6: Drop the box action

```
(:action drop_box
    :precondition (and
        (not (box_at_start))
        (robot_charged)
        (robot_carries_box)
        (robot_at_goal)
        (not(robot_at_start)))
    :effect (and
        (not (robot_carries_box))
        (not(robot_charged))
        )
)
```

The following actions involve the ability of a robot to open and close a box. In *Listing* 7 the action to open a box is shown. A robot can always open a box as long as it is charged.

Listing 7: Open the box action

```
(:action open_box
    :precondition (and
        (robot_charged)
    )
    :effect (and
        (not (box_closed))
        )
)
```

In *Listing* 8 the action to close a box is presented. A robot can close a box only if it is open and as long as it is charged. This action causes the robot to be discharged.

Listing 8: Close the box action

```
(:action close_box
    :precondition (and
        (not (box_closed))
        (robot_charged)
        )
    :effect (and
        (box_closed)
        (not (robot_charged))
        )
)
```

The following actions involve cooperation between two robots in order to carry heavy boxes.

In *Listing* 9 the action that allow a big box to be placed at the start location is shown. It can be imagined that this action is performed by a different kind of robot capable of lifting heavy boxes and it can bring a big box to the start location without any precondition. The "robots_composed" predicate indicates that when the big box is at the start location, there are no robots that are composed to form a bigger one.

Listing 9: Bring a big box to the warehouse action

```
(:action bring_bigbox
    :effect (and
        (bigbox_at_start)
        (not (robots_composed))
     )
)
```

When a big box is at the start location, a robot calls another one. This is what indicates the predicate "robot_call", this cause another robot to come and help the first one.

Listing 10: Call another robot action

```
    (:action call_another_robot
        :precondition (and
            (robot_charged)
            (bigbox_at_start)
            (not (robots_composed))
        )
        :effect (and
            (robot_call)
        )
    )
```

After a robot call, two robots can perform a composition to form a bigger robot that can carry a big box. The action is shown in *Listing* 11. This action cause the "robot_call" to be false.

Listing 11: Perform robot composition action

```
    (:action perform_robot_composition
        :precondition (and
            (robot_charged)
            (bigbox_at_start)
            (robot_call)
        )
        :effect (and
            (robots_composed)
            (not (robot_call))
            )
    )
```

Once two robots are composed together, they can put a big box in the goal position. The *Listing* 12 presents the action that allows this. As it is possible to see, the process to put a big box in the goal position is simpler than the one with one robot only. This because this was just to differentiate the case where two robots are needed, but it can be imagined that the process of putting a big box in the goal position is very similar to the one of a normal box.

Listing 12: Put a big box on the goal position action

```
    (:action put_bigbox_to_goal
        :precondition (and
            (robot_charged)
            (robots_composed)
            (bigbox_at_start)
        )
        :effect (and
            (not (bigbox_at_start))
         )
```

```
    )
```

When a big box is put in the goal position, the two robots completed the task and can be divided again performing a decomposition action, shown in *Listing* 13. This action cause the robots to be discharged.

Listing 13: Perform the robot decomposition action

```
    (:action perform_robot_decomposition
        :precondition (and
            (robot_charged)
            (not (bigbox_at_start))
            (robots_composed)
            (not (robot_call))
        )
        :effect (and
            (not (robots_composed))
            (not (robot_charged))
        )
    )
```

2.2 K

For the implementation in K language, the domain has been simplified. Here a robot simply picks a box from a start location and put it in a goal location. The code for such part is shown in *Listing* 14.

Listing 14: K code for the simplified version of the amazon racking robots domain

```
    fluents:
        box_at_start.
        robot_at_start.
        robot_at_goal.
        robot_charged.
        robot_carries_box.

    actions:
            go_to_start.
            pick_box.
            go_to_goal.
            drop_box.

    always:
            executable go_to_start if robot_charged.
            caused robot_at_start after go_to_start.
            caused -robot_at_goal after go_to_start.
```

```
        executable pick_box if robot_at_start.
        executable pick_box if robot_charded.
        executable pick_box if box_at_start.
        caused robot_carries_box after pick_box.

        executable go_to_goal if robot_charged.
        executable go_to_goal if box_at_start.
        executable go_to_goal if robot_carries_box.
        caused -box_at_start after go_to_goal.
        caused robot_at_goal after go_to_goal.
        caused -robot_at_start after go_to_goal.

        executable drop_box if -box_at_start.
        executable drop_box if -robot_at_start.
        executable drop_box if robot_at_goal.
        executable drop_box if robot_charged.
        executable drop_box if robot_carries_box.
        caused -robot_carries_box after drop_box.
        caused -robot_charged after drop_box.

initially:
        total box_at_start.
        robot_at_start.
        -robot_at_goal.
        total robot_charged.
        total robot_carries_box.


goal: -robot_carries_box? (4)
```

3 Experiments and results

In this section results during the testing phase are presented. The commands used to run the tests are shown in section 4

3.1 PDDL

For this part clingo, eclingo, plasp and quasp have been used along with the following encodings:

- `sequential-horizon.uurev.general.eclingo`;
- `sequential-horizon.uurev.eclingo`.

For all the tests the horizons tested have been from 1 to 5.

3.1.1 Testing with clingo

Below some screenshots of the tests executed are presented. It is possible to see that the computation with clingo is quite fast even with an horizon of 5.

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ ./plasp translate code/amazonrackingrobots.pddl | clingo 0 reversibility-asp-elp-aspq/sequential-horizon.uurev.lp -c horizon=1 -
clingo version 5.4.0
Reading from ...p-elp-aspq/sequential-horizon.uurev.lp ...
Solving...
UNSATISFIABLE

Models       : 0
Calls        : 1
Time         : 0.008s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.007s
(eclingoenv) [tobia@tobiac ATAI2_Project]$ ./plasp translate code/amazonrackingrobots.pddl | clingo 0 reversibility-asp-elp-aspq/sequential-horizon.uurev.lp -c horizon=2 -
clingo version 5.4.0
Reading from ...p-elp-aspq/sequential-horizon.uurev.lp ...
Solving...
Answer: 1
chosen("closebox") plan("go_to_charge",1) plan("openbox",2)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.010s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.007s
(eclingoenv) [tobia@tobiac ATAI2_Project]$ ./plasp translate code/amazonrackingrobots.pddl | clingo 0 reversibility-asp-elp-aspq/sequential-horizon.uurev.lp -c horizon=3 -
clingo version 5.4.0
Reading from ...p-elp-aspq/sequential-horizon.uurev.lp ...
Solving...
Answer: 1
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3)
Answer: 2
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3)
Answer: 3
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3)
SATISFIABLE

Models       : 3
Calls        : 1
Time         : 0.011s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.010s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 1: test using clingo with horizon=1,2,3

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ ./plasp translate code/amazonrackingrobots.pddl | clingo 0 reversibility-asp-elp-aspq/sequential-horizon.uurev.lp -c horizon=4 -
clingo version 5.4.0
Reading from ...p-elp-aspq/sequential-horizon.uurev.lp ...
Solving...
Answer: 1
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("go_to_charge",3) plan("openbox",4)
Answer: 2
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("openbox",4)
Answer: 3
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("openbox",4)
Answer: 4
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("openbox",4)
Answer: 5
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("go_to_charge",4)
Answer: 6
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("go_to_charge",4)
Answer: 7
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("go_to_charge",4)
SATISFIABLE

Models       : 7
Calls        : 1
Time         : 0.010s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.008s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 2: test using clingo with horizon=4

```
Answer: 3
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("go_to_charge",4) plan("openbox",5)
Answer: 4
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("openbox",4) plan("openbox",5)
Answer: 5
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("go_to_charge",4) plan("openbox",5)
Answer: 6
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("openbox",4) plan("openbox",5)
Answer: 7
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("go_to_charge",4) plan("openbox",5)
Answer: 8
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("openbox",4) plan("openbox",5)
Answer: 9
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("go_to_charge",3) plan("openbox",4) plan("go_to_charge",5)
Answer: 10
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("go_to_charge",4) plan("go_to_charge",5)
Answer: 11
chosen("closebox") plan("go_to_charge",1) plan("go_to_charge",2) plan("openbox",3) plan("openbox",4) plan("go_to_charge",5)
Answer: 12
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("go_to_charge",4) plan("go_to_charge",5)
Answer: 13
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("go_to_charge",3) plan("openbox",4) plan("go_to_charge",5)
Answer: 14
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("go_to_charge",4) plan("go_to_charge",5)
Answer: 15
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("openbox",3) plan("openbox",4) plan("go_to_charge",5)
Answer: 16
chosen("closebox") plan("go_to_charge",1) plan("openbox",2) plan("closebox",3) plan("go_to_charge",4) plan("openbox",5)
Answer: 17
chosen("perform_robot_decomposition") plan("go_to_charge",1) plan("bring_bigbox",2) plan("call_another_robot",3) plan("perform_robot_composition",4) plan("put_bigbox_to_goal",5)
Answer: 18
chosen("perform_robot_decomposition") plan("bring_bigbox",1) plan("go_to_charge",2) plan("call_another_robot",3) plan("perform_robot_composition",4) plan("put_bigbox_to_goal",5)
Answer: 19
chosen("drop_box") plan("go_to_charge",1) plan("go_to_start",2) plan("bring_box",3) plan("pick_box",4) plan("go_to_goal",5)
Answer: 20
chosen("drop_box") plan("go_to_charge",1) plan("bring_box",2) plan("go_to_start",3) plan("pick_box",4) plan("go_to_goal",5)
Answer: 21
chosen("drop_box") plan("bring_box",1) plan("go_to_charge",2) plan("go_to_start",3) plan("pick_box",4) plan("go_to_goal",5)
SATISFIABLE

Models       : 21
Calls        : 1
Time         : 0.008s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.006s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 3: test using clingo with horizon=5

3.1.2 Testing with eclingo

Below some screenshots of the testing executed with eclingo are presented. It is possible to see that the time required for the computation using `sequential-horizon.uurev.general.eclingo` is much larger than the one required with clingo. When using `sequential-horizon.uurev.eclingo`, the results are computed much faster.

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ eclingo -n 0 -c horizon=1 amazonrackingrobots.pddl.lp reversibility-asp-elp-aspq/sequential-horizon.uurev.general.eclingo
eclingo version 0.2.0
Solving...
UNSATISFIABLE

Elapsed time: 0.571921 s
(eclingoenv) [tobia@tobiac ATAI2_Project]$ eclingo -n 0 -c horizon=2 amazonrackingrobots.pddl.lp reversibility-asp-elp-aspq/sequential-horizon.uurev.general.eclingo
eclingo version 0.2.0
Solving...
Answer: 1
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("openbox", 2) }
SATISFIABLE

Elapsed time: 6.900471 s
(eclingoenv) [tobia@tobiac ATAI2_Project]$ eclingo -n 0 -c horizon=3 amazonrackingrobots.pddl.lp reversibility-asp-elp-aspq/sequential-horizon.uurev.general.eclingo
eclingo version 0.2.0
Solving...
Answer: 1
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("openbox", 2) }
Answer: 2
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("openbox", 3) }
Answer: 3
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) }
SATISFIABLE

Elapsed time: 77.274998 s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 4: test using eclingo with horizon=1,2,3

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ eclingo -n 0 -c horizon=4 amazonrackingrobots.pddl.lp reversibility-asp-elp-aspq/sequential-horizon.uurev.general.eclingo
eclingo version 0.2.0
Solving...
Answer: 1
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 3) }
Answer: 2
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) }
Answer: 3
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) }
Answer: 4
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) }
Answer: 5
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 3) } &k{ plan("openbox", 4) }
Answer: 6
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("openbox", 2) } &k{ plan("openbox", 4) }
Answer: 7
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 2) }
SATISFIABLE

Elapsed time: 826.776787 s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 5: test using clingo with horizon=4

```
&k{ chosen("drop_box") } &k{ plan("bring_box", 2) } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_goal", 5) } &k{ plan("go_to_start", 3) } &k{ plan("pick_box", 4) }
Answer: 3
&k{ chosen("drop_box") } &k{ plan("bring_box", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_goal", 5) } &k{ plan("go_to_start", 3) } &k{ plan("pick_box", 4) }
Answer: 4
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 5) }
Answer: 5
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 3) } &k{ plan("openbox", 4) } &k{ plan("openbox", 5) }
Answer: 6
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 4) }
Answer: 7
&k{ chosen("perform_robot_decomposition") } &k{ plan("bring_bigbox", 2) } &k{ plan("call_another_robot", 3) } &k{ plan("go_to_charge", 1) } &k{ plan("perform_robot_composition", 4) } &k{ plan("put_bigbox
_to_goal", 5) }
Answer: 8
&k{ chosen("perform_robot_decomposition") } &k{ plan("bring_bigbox", 1) } &k{ plan("call_another_robot", 3) } &k{ plan("go_to_charge", 2) } &k{ plan("perform_robot_composition", 4) } &k{ plan("put_bigbox
_to_goal", 5) }
Answer: 9
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("openbox", 2) } &k{ plan("openbox", 4) } &k{ plan("openbox", 5) }
Answer: 10
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 2) } &k{ plan("openbox", 5) }
Answer: 11
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 2) } &k{ plan("openbox", 4) }
Answer: 12
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 3) } &k{ plan("go_to_charge", 4) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 2) }
Answer: 13
&k{ chosen("closebox") } &k{ plan("closebox", 3) } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 2) } &k{ plan("openbox", 5) }
Answer: 14
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) } &k{ plan("openbox", 5) }
Answer: 15
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) }
Answer: 16
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 3) } &k{ plan("openbox", 5) }
Answer: 17
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 2) } &k{ plan("go_to_charge", 4) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 3) }
Answer: 18
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) } &k{ plan("openbox", 5) }
Answer: 19
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 4) }
Answer: 20
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 4) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) } &k{ plan("openbox", 5) }
Answer: 21
&k{ chosen("closebox") } &k{ plan("go_to_charge", 1) } &k{ plan("go_to_charge", 4) } &k{ plan("go_to_charge", 5) } &k{ plan("openbox", 2) } &k{ plan("openbox", 3) }
SATISFIABLE

Elapsed time: 9744.013463 s
(eclingoenv) [tobia@tobiac ATAI2_Project]$
```

Figure 6: test using clingo with horizon=5

3.1.3 Testing with qasp

Below some screenshots of the testing executed using qasp are presented. The tests with horizon=4 and horizon=5 have been terminated by the user after several hours as the time required for the computation (not reported by qasp) was very large.

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ reversibility-asp-elp-aspq/./pddl-to-qasp-uurev.py code/amazonrackingrobots.pddl 2 > amazonrackingrobots.pddl.qasp
(eclingoenv) [tobia@tobiac ATAI2_Project]$ java -jar reversibility-asp-elp-aspq/qasp.jar -n -1 -m -f chosen,plan amazonrackingrobots.pddl.qasp
SAT
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2)}
(eclingoenv) [tobia@tobiac ATAI2_Project]$ reversibility-asp-elp-aspq/./pddl-to-qasp-uurev.py code/amazonrackingrobots.pddl 3 > amazonrackingrobots.pddl.qasp
(eclingoenv) [tobia@tobiac ATAI2_Project]$ java -jar reversibility-asp-elp-aspq/qasp.jar -n -1 -m -f chosen,plan amazonrackingrobots.pddl.qasp
SAT
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("go_to_charge",3)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("openbox",3)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("openbox",3)}
(eclingoenv) [tobia@tobiac ATAI2_Project]$ reversibility-asp-elp-aspq/./pddl-to-qasp-uurev.py code/amazonrackingrobots.pddl 4 > amazonrackingrobots.pddl.qasp
(eclingoenv) [tobia@tobiac ATAI2_Project]$ java -jar reversibility-asp-elp-aspq/qasp.jar -n -1 -m -f chosen,plan amazonrackingrobots.pddl.qasp
SAT
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",3), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("openbox",3), plan("go_to_charge",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("openbox",3), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("go_to_charge",3), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("openbox",3), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("go_to_charge",3), plan("go_to_charge",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("openbox",3), plan("go_to_charge",4)}
```

Figure 7: test using quasp with horizon=2,3,4

```
(eclingoenv) [tobia@tobiac ATAI2_Project]$ reversibility-asp-elp-aspq/./pddl-to-qasp-uurev.py code/amazonrackingrobots.pddl 5 > amazonrackingrobots.pddl.qasp
(eclingoenv) [tobia@tobiac ATAI2_Project]$ java -jar reversibility-asp-elp-aspq/qasp.jar -n -1 -m -f chosen,plan amazonrackingrobots.pddl.qasp
SAT
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",3), plan("openbox",2), plan("openbox",4), plan("openbox",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",3), plan("go_to_charge",5), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("openbox",2), plan("openbox",3), plan("openbox",4), plan("openbox",5)}
{chosen("drop_box"), plan("go_to_charge",1), plan("bring_box",3), plan("go_to_start",2), plan("go_to_goal",5), plan("pick_box",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",4), plan("go_to_charge",5), plan("openbox",2), plan("openbox",3)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",3), plan("go_to_charge",5), plan("openbox",2), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("openbox",3), plan("openbox",4), plan("openbox",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",3), plan("openbox",4), plan("openbox",5)}
{chosen("drop_box"), plan("go_to_charge",2), plan("bring_box",1), plan("go_to_start",3), plan("go_to_goal",5), plan("pick_box",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",5), plan("openbox",3), plan("openbox",4)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",4), plan("go_to_charge",5), plan("openbox",3)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",3), plan("go_to_charge",4), plan("openbox",2), plan("openbox",5)}
{chosen("perform_robot_decomposition"), plan("go_to_charge",2), plan("bring_bigbox",1), plan("call_another_robot",3), plan("perform_robot_composition",4), plan("put_bigbox_to_goal",5)}
{chosen("perform_robot_decomposition"), plan("go_to_charge",1), plan("bring_bigbox",2), plan("call_another_robot",3), plan("perform_robot_composition",4), plan("put_bigbox_to_goal",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",4), plan("openbox",2), plan("openbox",3), plan("openbox",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",3), plan("go_to_charge",4), plan("go_to_charge",5), plan("openbox",2)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",3), plan("go_to_charge",4), plan("openbox",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",2), plan("go_to_charge",4), plan("openbox",3), plan("openbox",5)}
{chosen("closebox"), plan("go_to_charge",1), plan("go_to_charge",4), plan("openbox",2), plan("openbox",5), plan("closebox",3)}
```

Figure 8: test using quasp with horizon=5

4 How to run the code

To run the code, the following requirements have to be satisfied:

- eclingo version 0.2.0;
- clingo version 5.4.0;
- plasp version 3.1.1;
- qasp version 0.1.

In Listings 15 and 16 the commands used to run the tests with clingo and eclingo respectively are shown. In the commands, n is an integer number indicating the horizon, while `sequential-horizon.uurev.eclingo` can be also substituted with `sequential-horizon.uurev.general.eclingo`.

Listing 15: clingo test command

```
./plasp translate ../amazonrackingrobots.pddl | clingo 0 .../
    sequential-horizon.uurev.lp -c horizon=n -
```

Listing 16: eclingo test command

```
./plasp translate ../amazonrackingrobots.pddl | eclingo -n 0 -c
    horizon=n ../sequential-horizon.uurev.general.eclingo
```

To run the tests with qasp, first the PDDL file needs to be translated, to do that the command shown in *Listing* 17 can be used. Note that here the n is an integer number that specifies the horizon.

Listing 17: translate pddl file into qasp file

```
reversibility-asp-elp-aspq/./pddl-to-qasp-uurev.py code/
    amazonrackingrobots.pddl n > amazonrackingrobots.pddl.qasp
```

Then to execute the test with qasp, the command shown in *Listing* 18 can be used. This command is to print all models and filter to chosen and plan, to prin a model instead of all models, remove -n -1 options.

Listing 18: execute qasp test

```
java -jar reversibility-asp-elp-aspq/qasp.jar -n -1 -m -f chosen,
    plan amazonrackingrobots.pddl.qasp
```