Universität des Saarlandes
Fachrichtung 6.2 – Informatik
J. Siekmann, S. Autexier, C. Benzmüller, C.E. Brown
C. Hahn

4th Practical Assignment in Artificial Intelligence (SS 2005)

Issued: June 27, 2005 Due: July 11, 2005

This practical assignment should make you familiar with the planning system UCPOP

Literature: Russell/Norvig, *Artificial Intelligence: A Modern Approach*, Chapter 11
Barrett, Christianson, Friedman, Kwok, Golden, Penberthy, Sun, Weld, *UCPOP User's manual (Version 4.0)*, Chapters 3,4 (see web-page of the exercises)

The planning domain is a robot who can move around between different rooms. Each room has a unique position (x, y). Furthermore there are objects in the room, some of which can be moved, some not.

Objects

The robots world contains two movable objects: A *Box* and a *Stick*. The movable objects are inside some rooms (i.e., movable objects have a unique position (x, y)). There can be more than one object in a room, but the robot can only carry a **single** movable object at the time.

Robot

The robot has one arm which allows it to carry a single movable object. In order to grab an object, its arm must be empty. Furthermore, it must be in the same room as the object and execute an action get-object.

If the robot moves around holding an object, then the object moves with it; i.e., the object is always in the same room as the robot is.

The robot can move between the different rooms. To simplify matters, we assume there is an operator move-to, which allows the robot to move from the current room to an arbitrary other room.

Climbing on the Box

The robot can climb on the box, if the box is in the same room than it is and it is not already standing on the box. Furthermore, it cannot climb on the box, if it is carrying the box. While the robot stands on the box, it cannot leave the room.

The lamp

In one of the rooms there is a lamp, which cannot be moved. Initially the lamp is off. Since the robot is too small to turn the lamp on while standing on the floor, it must be creative. It has the following possibilities:

- It can jump in order to turn the lamp on. But to do so, it must be in the same room as the lamp and not carrying anything.
- It can climb on the box in order to turn the lamp on. To do so the robot must be standing on the box, not carry anything and the lamp must be in the same room as the box.
- Finally, it can use the stick to reach the lamp and turn it on. To do so it must hold the stick and be in the same room as the lamp.

Planning Operators

We give you a list of actions, which the robot can take in order to achieve a given goal. These actions are given in the file `robotworld.lisp` which also includes additional information. You can find that file on the web-page containing the exercises. Please note the comments at the beginning of the file: they provide you with details about how to represent a state of the robot world when you define the preconditions of the actions. In this representation, rooms are given by symbolic names like `x1-y8` instead of pairs $(\mathtt{x}, \mathtt{y})$. Doing so we can concentrate on the interesting rooms, such as the room in which the robot is initially, the room where the lamp is, etc.

- `move-to`: This operator has two parameters: the room the robot is currently in and the room the robot wants to move to. To execute that action, the robot must actually be in the first room. We assume the robot can reach any room it wants to go to.
- `get-object` and `drop-object`: With these operators the robot can grab or release objects. Both have the same parameters: the object the robot wants to grab or to release and the current room. Both actions can only be executed under the conditions sketched above.
- `climb-box` and `descend-box`: With these operators the robot can climb on the box or descend from the box, if the necessary preconditions are met. These operators only have the current room as a parameter.
- `jump-light-on` `jump-light-off`: With these operators the robot can try to turn the lamp on (resp. off) by jumping. These operators only have the current room as a parameter.
- `switch-light-on` and `switch-light-off`: With these operators the robot can try to turn the lamp on (resp. off) if it is standing on the box and it is in the same room than the lamp is. These operators only have the current room as a parameter.
- `stick-light-on` and `stick-light-off`: With these operators the robot can try to turn the lamp on (resp. off) by using its stick. To do so, the robot must hold the stick and be in the same room as the lamp is. These operators only have the current room as a parameter.

Starting UCPOP

First start lisp (e.g., `/home/stud/kiss/bin/lisp` or your version of cmucl18e installed). Then load the file

`/home/stud/kiss/ucpop-4.1/loader.lisp`

by `(load ''/home/stud/kiss/ucpop-4.1/loader.lisp'')` and then execute the functions

`(load-ucpop)` and `(in-package ''UCPOP'')`.

Finally, load your version of `robotworld.lisp`. The original version of that file–which we provide you– contains the above, incomplete actions, and the following planning problems:

```
simple-move
turn-on-light
stick-and-light
get-on-box
box-and-light
complex
```

In order to generate a plan using UCPOP, you can use the function `bf-control` or its iterative deepening version `ibf-control`. Example: `(bf-control 'simple-move)`

Exercise 4.1 (Definition of the Planning Actions) (66 P)

Define the preconditions and effects of the actions for the robotworld domain in your copy of the file `robotworld.lisp`. Make sure to reload the file each you changed something. Check that your actions fulfil all conditions of the robot world. Furthermore, the robot shall not execute redundant actions, like for instance to try to climb on the box if it is already standing on it.

Please comment your changes directly in your `robotworld.lisp` file. Your comments should make your lisp-code more comprehensible and explain the mode of operation of algorithms you may have inserted. For each correct and adequately described action you get (6 P).

Exercise 4.2 (Test Runs) (34+20 P)

Using your file `robotworld.lisp` use UCPOP to generate plans for the following problems. There are problems where you will have to play around with the settings of the planner in order to obtain a plan. In case you nevertheless cannot solve a problem, turn in the result of the planner.

1. `simple-move` (6 P)
2. `turn-on-light` (7 P)
3. `stick-and-light` (7 P)
4. `get-on-box` (7 P)
5. `box-and-light` (7 P)
6. **`complex` (Bonus, 20 P)**