Lab class: planning

Steven Schockaert

Week 6

At the end of the exercise class, please post a message on the discussion board on learning central, indicating which parts of the assignment you have been able to solve and what problems you faced. Feedback in the subsequent lecture will address the issues that were highlighted in this way.

GPlan

For this lab, we will use GPlan, a STRIPS planner implemented in java. To get started, download `lab4-source.zip` from learning central. To run the planner on one of the provided examples, on Windows you can use the following command:

```
java -cp dist\GPlan.jar cz.matfyz.sykora.planning.Planner examples\hanoi.plan
```

On unix based systems you can use:

```
./run examples/hanoi.plan
```

To encode a planning problem, the system expects a text file which contains the definition of the possible actions, the set of conditions which are satisfied in the initial state and the set of conditions that need to be satisfied in the goal state. For example, in the case of the towers of Hanoi problem (`examples/hanoi.plan`), there is only one action:

```
move ::  bigger(X, Y), empty(X), empty(Y), on(X, Z) => empty(Z), not empty(Y), not
on(X, Z), on(X, Y).
```

where `move` is the name of the action, `bigger(X, Y)` $\wedge$ `empty(X)` $\wedge$ `empty(Y)` $\wedge$ `on(X, Z)` is the pre-condition and `empty(Z)` $\wedge \neg$`empty(Y)` $\wedge \neg$`on(X, Z)` $\wedge$ `on(X, Y)` is the effect. Upper-case arguments such as `X` represent variables. When describing actions it is also possible to refer to specific constants, which are denoted using lower-case arguments. For example, in a blocks world, a possible action could be:

```
takeFromTable ::  block(X), on(X, table) => holding(X), not on(X,table).
```

where `table` is a constant, referring to a specific object. The conditions which are true in the initial state are listed on separate lines, e.g. in the towers of Hanoi problem, the following line encodes the fact that `on(k1,k2)` is true initially:

```
on(k1,k2).
```

Finally, the conditions which need to be true in the goal state are specified by using the keyword goal, e.g. in the towers of Hanoi problem, the following line encodes the fact that `on(k1,k2)` needs to hold in the goal state:

```
goal on(k1,k2).
```

The syntax which is used to encode planning problems is explained in more detail on GPlan website[1] and in the `README.rdoc` file included in the archive.

Assignment

1. To get familiar with the syntax, start by implementing the blocks world example from the lecture notes (slides 2-7).
2. Encode the following river crossing puzzle as a planning problem, and solve it using GPlan (from Wikipedia[2]):

   > A man and a woman of equal weight, together with two children, each of half their weight, wish to cross a river using a boat which can only carry the weight of one adult.

   Hint: encode the problem using four actions, i.e. a single person moving to the other side, two people moving to the other side, a single person coming back and two people coming back.
3. Encode the following river crossing puzzle as a planning problem, and solve it using GPlan (from Wikipedia[3]):

   > Three married couples must cross a river using a boat which can hold at most two people, subject to the constraint that no woman can be in the presence of another man unless her husband is also present.

   Hint: use separate actions to encode under what condition a given man can move to the other side alone, under which conditions a given woman can move to the other side alone, under which conditions two men can move to the other side together, etc. For example, a man can only go to the other side if (a) his wife is already on the other side, (b) all the other men are on the other side, or (c) his wife is moving to the other side with him. This corresponds to three separate actions, and three corresponding actions about moving back to the initial side. A full encoding of the problem setting in this way requires 16 different actions.

[1] https://github.com/ondrasej/GPlan
[2] `https://en.wikipedia.org/wiki/River_crossing_puzzle`
[3] `https://en.wikipedia.org/wiki/River_crossing_puzzle`