Exercise 10

Foundations of Agent Programming

Prof. Michael Thielscher, Sebastian Voigt

International Master Program in Computational Logic — summer term 2009

09.07.2009

Exercise 10.1 Repetition

(Chapter 3: "Planning")

Reconsider the *Logistics domain* (cf. Slides 3b/12ff). In this exercise, we consider an instance with the single vehicle $Plane$, the single object $Mail$ and three locations: $Dresden$, $Leipzig$, $Berlin$ and $Depot$.

a) Specify LTL formulas for the following statements:

φ_1: There should never be mail in the plane.

φ_2: The plane should never fly to Leipzig.

φ_3: If the first action of the plane is to fly to Leipzig, then it should load mail immediately afterwards.

b) Consider the following *preference formula* (cf. Slide 3b/23), where $\varphi_1, \varphi_2, \varphi_3$ are the formulas obtained in a):

$$\Phi = (\varphi_1 \mid (\varphi_2 \prec \varphi_3)) \quad \& \quad (\Diamond At(Plane, Depot) \Rightarrow \neg\varphi_2) \quad \& \quad \texttt{final}(At(Plane, Depot))$$

Assume that initially the plane is empty and in Berlin and that all actions used below have the expected effects. Which of the following two plans P_1 and P_2 would be preferred according to the weight of Φ? (For basic desire formulas you don't have to show every step of the weight calculation if you can determine the weight "intuitively".)

P_1: $S_0, S_1, S_2, S_3, S_3, \ldots$, where

$$S_1 = Do(Fly(Plane, Dresden), S_0)$$
$$S_2 = Do(Load(Mail, Plane), S_1)$$
$$S_3 = Do(Fly(Plane, Leipzig), S_2)$$

P_2: $S_0, S_1, S_2, S_3, S_3, \ldots$, where

$$S_1 = Do(Fly(Plane, Leipzig), S_0)$$
$$S_2 = Do(Load(Mail, Plane), S_1)$$
$$S_3 = Do(Fly(Plane, Depot), S_2)$$

c) Given Plan P_2 and Φ from b), calculate $\Phi''' = Progress^3(\Phi)$ and verify that the weight of Φ''' equals the weight of Φ.

Exercise 10.2 Repetition

(Chapter 3: "Planning")

In this exercise we first define a simplified variant of the *Logistics domain* which does not consider trucks. The task then is to provide an *Answer Set Program* for the simplified domain that is able to solve arbitrary problem instances, where the ASP encoding should be based on a description in the *Game Description Language*. We use the following fluents (cf. Slide 3b/12):

$At(o, l)$	$\hat{=}$	object o is at location l
$In(o, v)$	$\hat{=}$	object o is in vehicle v
$Loc(v, l)$	$\hat{=}$	vehicle v it at location l
$Req(o, l_1, l_2)$	$\hat{=}$	there is a request to move object o from location l_1 to location l_2

Vehicles will be considered to be players of the game. Thus in actions we omit the argument of the vehicle, as it will be passed via the $Does$ statement in the GDL:

$Load(o)$	$\hat{=}$	object o is loaded
$Unload(o)$	$\hat{=}$	object o is unloaded
$Fly(l)$	$\hat{=}$	fly to location l
$Noop$	$\hat{=}$	do nothing

Download the file `10.2.pl` from the course web page. It contains some domain definitions as well as an example initial state encoding for the ASP. Solve the following tasks:

a) Provide ASP clauses such that exactly one instance of `does(V,A,T)` is true at every time step `T` and thus every vehicle `V` does exactly one action `A` at `T`.

b) Define a predicate `terminal(T)` such that the predicate is true if no request is left at time step `T`. Assure that the predicate is true eventually and that no action is performed thereafter.

c) Define precondition axioms via the predicate `legal(V,A,T)` such that the predicate is true if for vehicle `V` it is legal to do action `A` at time step `T`.

d) Define effect axioms via the predicate `holds(F,T)` such that the predicate is true if fluent `F` holds at time step `T`.

e) Provide integrity constraints for the statements 1., 2. and 4. from Slide 3b/13.

f) Test your encoding with the domain instance specified in `10.2.pl`.