Architecture of a Controller that reasons with initial defaults, diagnosis of exogeneous actions, and planning with intention.

Rocio Gomez

March 20, 2019

1 The Controller.

Our robot only deals with reasoning at one abstract level. At this point we assume that exectuer can act and observe in the world at this level of abstraction too. We will add refinement later. The robot will reason with defaults, diganosis of exogeneous actions and planning with intention each time it has an interaction with the world execution of actions and the relevan observatinos have happened.

This controller is going to use the same ASP file for defaults, diagnosis and planning by running it three consecutive times using three different flags: *finding_defaults(step)*, *diagnosing(step)* and *planning(step)*. These flags, given as input to the ASP, allow different rules in the ASP to be triggered.

1.1 The physical domain

We will explain the architecture of the controller using the following domain as example:

The domain has four rooms located side by side *office1*, *office2*, *kitchen* and *library*) and connected. The robot, which we call *rob*, can move from one room to the next. A room that is *secure* can be *locked* or *unlocked*. The robot cannot move to or from a locked room; it can *unlock* a locked room. The domain objects can be located in any of the rooms. The robot can *pickup* an object if the robot and the object are in the same room, and it can *put down* an object that it is holding; the robot can only hold one object at a time. The domain includes two exogenous actions *exo_move* and *exo_lock*, one that can change the location of any object, and the other that can lock a secure room. The agent may or may not be aware of these exogenous action when they happen.

Through out this text we assume that the goal of the robot will be to put both books in the *library*. The goal will be reached when the location of both books are in the *library* and the robot is not holding any of them.

In our domain, by default, books are going to be located in *office1* or in *office2*, with preference of them being in *office1*. So by default a book will be in *office1*, and if not it

will be in *office2*. If it is known that a book is not in either one of these default locations, then they can be in the *library* or *kitchen* with equal probability.

At the initial state of the planning, the robot may or may not know the location of the books, or if the door of the *library* is locked. The first thing our robot does is to observe the value of those fluents that can be observed from its location, such is the location of the robot itslef, what it is holidng in its hand, what objects are or are not in its same location and, in the case of being in the *library* or in the *kitchen*, if the door to the *library* is locked or unlocked. Once it has observed the room, it will need to reason about defaults, diagnose observations and create a plan.

1.2 Example of the three phase resoning

Let us assume that our robot is in *office1*, with *book1* in its hand. Let us assume that the robot has not been given any information about the state of the domain and all it knows is its goal. After its initial observation the robot knows that it is located in *office1*, that *book1* is in its hand, and that *book2* is not in *office1*. With this information the controller goes through three stages of reasoning.

1. Finding defaults: To reason about fluents, the controller will run the ASP together with the history (so far only we have the initial obserations) and the flag *finding_defaults(0)*. In or example domain, during this phase the ASP concludes that *book2* is by default, in *office2*, since it is not in *office1*. The output of running this ASP will be of the form *defined_by_default(loc(book2,office2))*. This information will be given as input to the second phase of the reasoing together with the flag *diganosing(0)*.

   The following rule was included in the section of history rules in the AL description:

   $$holds(F,0) \quad \leftarrow \quad \begin{array}{l} defined\_by\_default(F). \\ not\ finding\_defaults(I). \end{array} \tag{1}$$

   So the ASP in the *diagnosing* or *planning* phase will consider that a fluent defined by default hold at 0.

2. Diagnosis: Now the robot, using its observations and the fluents that have been defined by default, will enter in the second phase of its reasoing. It will do two things:

   - Diagnosing unexpected current observations that are not consistent with past observations.
   - Choose a value of those fluents that either have not been observed, or that have not been defined by defaults.

   Diagnosing does not make sense to do in the first step, as there is not a past history to contrast it with. However, in other steps, if exogeneouse actions have been diagnosed to reconciliate unexpected observations.