Well, if we have a planner than we can subsume 'things I know' under 'model of the world'

We have a 'true' world model Mw, which we use to drive the simulator. Now besides Mw we need some base values, V, e.g. 'dying is bad'. V of course needs to be expressed in some language that can encode value and link it to productions in Mw.

So optimal learning state for Otopopo is M, where Otopopo's model of the world matches that of the simulator.

Of course I'm skirting the issue of what language M, Mw, and V are elements of. Call this language P, for reasons shortly to become clear.

P needs to be expressive enough that programmers can, in available resources, define Mw

P needs to be machine translatable to PDDL

And

Now, we have to ask, how does learning happen?

Each agent contains a model Mi , which is its internal model of Mw[[1]](#footnote-1).

The agent plans against M. Each PDDL action is notated with the relevant element(s) of Mi which produced it. As the agent executes the plan, if the plan doesn't work out, then it marks the associated elements of Mi as dubious. When the elements are sufficiently dubious, they're dropped.

Additionally, when an agent successfully executes a section of a plan it adds that section to Its Mi , initially marking it as quite dubious. Already present subsections have their dubiosness decreased. We probably need some caching mechanism by which he can forget chunks of plan not recently used.

Elements of the PDDL which came from V are not altered.

Now, how can Otopopo teach Lemonade?

Otopopo not only runs this algorithm for his own behavior, but for Lemonade, as a reactive planner. When he sees lemonade he looks for plan sections which Lemonade might plausibly be executing. As he observes more of Lemonades behavior, he can reduce the number of fitting plan sections until he knows Lemonade's intent. If the number of plausible actions further reduces to zero, then he can, effectively, ask Lemonade what she's doing. Lemonade can recite her current plan.

"I was going fishing, but when I got to the stream I didn't have the fish basket" (in PDDL)

Otopopo searches for a plan fragment which is successful and contains much of Lemonade's plan, where 'much of' is Levenshtien or some such. Otopopo can then utter this plan In P.

"In order to fish you must have the fish basket. The fish basket is usually in Bigmouth's hut"

So I don't believe there should be a distinction between your "I7" and your "pone". In:

I7 = Controlled English we teach the "world builder" to build from.

Pone = Controlled English we teach "bots" to think in

When I say "pone" i mean STRIPS

See - not at all what I mean.  
  
I am referring to "Cycorp's redesign of the concept of a prolog-like facts" and  3 under-stated consequences of that redesign.   Also,  in my minds eye I am starting at colored blocked architecture that was designed before my time called CYC.  To further complicate things .. I refer to some parts deprecated 12 years ago by Cycorp  ..  Reminiscent are present enough such as the Madlib modules.  Some have been neglected such as seemingly undersigned "Recursive Template Parser" a DCG-like  form of understanding system for English (it was the only parser that ever could be incrementally crowdsourced).   P.A.M. and S.A.M. modules written by Roger Schank are only 10% supported but ready to be written when you can recognize the STRIPS notations chosen by Cycorp.

OK, I'm baffled by this.

1. Interestingly, if you make this M instead, it predicts the widespread religious notion of detachment! [↑](#footnote-ref-1)