Department of Computing, Imperial College

491 Knowledge Representation Assessed coursework

Autumn 2017

Deadline: 24 November 2017

This assignment is primarily an exercise in computing answer sets, extensions of some simple Reiter default theories, and illustrations of the connections between them.

All of the examples are based on two classic problems in the AI literature concerning the formalisation of default rules of persistence ('laws of inertia'). The idea is that by doing the exercises you learn something about these classic examples and why they have been seen as presenting problems (or non-problems, depending on your point of view).

There is a lot of reading but, I hope, not too much writing. Some parts are optional—for your interest only, recommended, but not assessed. In most cases you calculate answer sets using the ASP solver `clingo`. You can find the executables, and some brief documentation, in the directory `/vol/lab/CLASP`. You simply type (Linux command line):

`/vol/lab/CLASP/clingo 0` *file*$_1$`.lp ...` *file*$_n$`.lp`

`clingo` calls the grounder `gringo` and then pipes its output to the ASP solver `clasp` automatically. The argument `0` specifies that you want all solutions.

It is often instructive to look at the output produced by `gringo`. Type

`/vol/lab/CLASP/clingo -t` *file*$_1$`.lp ...` *file*$_n$`.lp`

If you want to put the `clingo` output in a file for later editing, you need to use the Linux `>` redirection: `/vol/lab/CLASP/clingo 0` *file*$_1$`.lp ...` *file*$_n$`.lp >` *output_file*.

Question 0 (Recommended, not assessed. Do not submit!!)

You will need the following observation to simplify some of the calculations.

Let P be a (ground) normal logic program, let A be a ground atom, and let $L_1, \ldots, L_n$ be ground literals.

1) For any normal logic program P, $P \cup \{\, A \leftarrow L_1, \ldots, L_n \,\} \cup \{\, A \leftarrow \,\}$ and $P \cup \{\, A \leftarrow \,\}$ have the same (Herbrand) models.

2) For any normal logic program P, $P \cup \{\, A \leftarrow L_1, \ldots, L_n \,\} \cup \{\, A \leftarrow \,\}$ and $P \cup \{\, A \leftarrow \,\}$ have the same stable models (answer sets).

Try to prove this but do *NOT* include the proof in your submission. (It is very easy. For (1), look at $\mathrm{T}_{P \cup \{\, A \leftarrow L_1, \ldots, L_n \,\} \cup \{\, A \leftarrow \,\}}(X)$. (2) follows from (1) — compare the two reducts. I will show the details in the model answer.) This result can also be generalised, to cover the case where the body of one clause subsumes the body of another clause with the same head allowing the program to be simplified without changing the answer sets. You don't need the more general version for this exercise.

Kautz's Stolen Car problem

Henry A. Kautz. The logic of persistence. In *Proc. National Conference on Artificial Intelligence (AAAI)*, pp 401–405, Philadelphia, 1986. Morgan Kaufmann Publishers, Inc.

A car is known (observed) to be in a car park at time 1. At some future time n, it is known (observed) not to be in the car park. Assuming some form of default persistence ('inertia'), where is the car at times between 1 and n?

Question 1 Let $p(t)$ represent that the car is in the car park at (integer) time t (and $\neg p(t)$ to represent that it is not).

Suppose that facts, positive and negative, persist forwards in time by default, as expressed by the following extended logic program P_{f}:

$$p(t{+}1) \;\leftarrow\; p(t),\ \text{not}\ \neg p(t{+}1)$$
$$\neg p(t{+}1) \;\leftarrow\; \neg p(t),\ \text{not}\ p(t{+}1)$$

This is shorthand for a set of ground clauses, for t ranging over integers from 1 to some $n{-}1$. Henceforth I'll refer to n as the 'maximum time' for the example.

Use 'splitting sets' to determine the answer sets of $P_{\mathrm{f}} \cup \mathit{Obs}$ for the following sets of observations Obs:

$$\{\, p(1),\ \neg p(3) \,\} \quad \text{and} \quad \{\, p(1),\ p(3) \,\} \quad \text{and} \quad \{\, p(2) \,\}$$

Here it makes sense to take 'maximum time' as 3.

Now, just for the practice, check your answers by computing the answer sets for these three problems using `clingo`. You will find the persistence rules in `clingo` syntax in the file `kautz_forward.lp` in `/vol/lab/CLASP/CWK-491-2017`. (The files are read-only so you will need to make your own copies if you want to edit them.) Type

```
/vol/lab/CLASP/clingo 0 kautz_forward.lp kautz_obs.lp
```

`kautz_obs.lp` is a `clingo` input file (also supplied) specifying the observations.

If you want to run the exercise for other values of 'maximum time', say 5, type:

```
/vol/lab/CLASP/clingo 0 --const maxT=5 kautz_forward.lp kautz_obs.lp
```

(This is just for your own interest. Do not submit the `clingo` outputs for this question, nor any splitting set calculations for maximum time greater than 3.)

Question 2 Let $P_{\rm b}$ be the following extended logic program, for t ranging over integers from 1 to some $n{-}1$.

$$p(t) \leftarrow p(t{+}1),\ \text{not}\ \neg p(t)$$
$$\neg p(t) \leftarrow \neg p(t{+}1),\ \text{not}\ p(t)$$

You can read $P_{\rm b}$ as expressing a kind of 'backwards persistence' of positive and negative facts over time. Some people simply cannot accept that 'backwards persistence' makes any sense. Causality, so they say, goes in one direction: a fact $p(t{+}1)$ cannot 'cause' another fact $p(t)$ at an earlier time t. I don't want to get into that here. Neither of the clauses in $P_{\rm b}$ say anything about causality. But anyway: notice that we can read $P_{\rm b}$ not as expressing some kind of causality in the car parks world, but simply as relating beliefs about where the car is at time t with beliefs about where the car is at time $t{+}1$. $P_{\rm b}$ then seems non-controversial (at least to me). Be that as it may.

Compute the answer sets of $P_{\rm f} \cup P_{\rm b} \cup \mathit{Obs}$ for the following sets of observations Obs:

$$\{\, p(1),\ \neg p(3) \,\} \quad \text{and} \quad \{\, p(1),\ p(3) \,\} \quad \text{and} \quad \{\, p(2) \,\}$$

As in Question 1, do this by hand using 'splitting sets'. I am sorry about the amount of writing but I believe it is instructive to do these calculations by hand. You will need to use Question 0 or you will not find any splitting sets. (You can also try the approximation method of Question 2 but *do not submit it*. Too much writing.)

Add a *brief* commentary to explain the significance of the answers.

As before you can use `clingo` to check your calculations. You will find the clauses $P_{\rm f} \cup P_{\rm b}$ in the file `kautz_forward_back.lp`. It is worth trying it for other values of 'maximum time', say 5. Do *not* submit the `clingo` outputs.

Question 3 In the literature, default persistence ('laws of inertia' or 'frame axioms') are often expressed using Reiter default logic. Consider this (typical) pair:

$$F = \{\ \begin{array}{l} \neg ab(t,t{+}1) \rightarrow (\, p(t) \rightarrow p(t{+}1)\,), \\ \neg ab(t,t{+}1) \rightarrow (\, \neg p(t) \rightarrow \neg p(t{+}1)\,) \end{array}\ \} \qquad D_{\neg ab} = \{\ \frac{:\, \neg ab(t,t{+}1)}{\neg ab(t,t{+}1)}\ \}$$

I have written F (for 'frame axioms') in this form to emphasise that F is a set of *formulas* (in first-order logic, or as here, in propositional logic with ground instances for t ranging over integers between 1 and $n{-}1$).

Temporal reasoning problems are then formulated as default theories $(D_{\neg ab}, F \cup W)$ where W is some set of formulas expressing effect axioms, observations, and other (non-temporal) features of the domain being modelled. Usually, of course, there are several time-varying facts or 'fluents' to deal with, and the default persistence/frame theory takes the form

$$F = \{\ \begin{array}{l} \neg ab(f,t,t{+}1) \rightarrow (\, f(t) \rightarrow f(t{+}1)\,), \\ \neg ab(f,t,t{+}1) \rightarrow (\, \neg f(t) \rightarrow \neg f(t{+}1)\,) \end{array}\ \} \qquad D_{\neg ab} = \{\ \frac{:\, \neg ab(f,t,t{+}1)}{\neg ab(f,t,t{+}1)}\ \}$$

for f ranging over every time-varying fact or 'fluent' of interest. Since the Stolen Car examples need only one such 'fluent' p (for 'parked') I have omitted the extra argument in ab for simplicity. But see the 'Yale Shooting Problem' exercises later.

(a) Show that the following are both extensions of $(D_{\neg ab},\ F \cup \{\, p(1), \neg p(3) \,\})$.

$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3) \,\} \cup \{\, \neg ab(1,2), ab(2,3) \,\})$$
$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3) \,\} \cup \{\, ab(1,2), \neg ab(2,3) \,\})$$

Do this by hand. Construct the reduct and then compute the relevant closure.

(b) (Recommended, not assessed. Do not submit!)

Are there any other extensions of $(D_{\neg ab},\ F \cup \{\, p(1), \neg p(3) \,\})$? One way to investigate it is as follows.

(i) First prove that if E is an extension of $(D_{\neg ab}, W)$ (any W) then either $ab(t,t{+}1) \in E$ or $\neg ab(t,t{+}1) \in E$. (Very easy! Suppose $ab(t,t{+}1) \notin E$. Then ...)

(ii) Now prove that there is no extension E of $(D_{\neg ab},\ F \cup \{\, p(1), \neg p(3) \,\})$ such that $ab(t,t{+}1) \in E$ and $\neg ab(t,t{+}1) \in E$. (Hint, to save work: extensions of Reiter default theories are always *minimal*, and you know from part (a) something about extensions of $(D_{\neg ab}, F \cup \{\, p(1), \neg p(3) \,\})$.)

(iii) It just remains to check whether there are any extensions of $(D_{\neg ab},\ F \cup \{\, p(1), \neg p(3) \,\})$ of the form

$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3) \,\} \cup \{\, ab(1,2), ab(2,3) \,\} \cup X)$$
$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3) \,\} \cup \{\, \neg ab(1,2), \neg ab(2,3) \,\} \cup X)$$

where X is to be determined, but does not contain any instances of ab literals.

Investigate this. (Show there are no such extensions.) This is for your own interest. Do not submit this.

(c) (For this part, there is something to submit — see below)

Since the defaults $D_{\neg ab}$ have the same meaning as the extended logic program clauses $\neg ab(t,t{+}1) \leftarrow \text{not}\ ab(t,t{+}1)$, you might think that the default theory $(D_{\neg ab}, F \cup \mathit{Obs})$ is equivalent, assuming Obs is a set of facts (ground literals), to the logic program $P_F \cup P_{\neg ab} \cup \mathit{Obs}$ where

$$P_F = \{\ \begin{array}{l} p(t{+}1) \leftarrow p(t),\ \neg ab(t,t{+}1), \\ \neg p(t{+}1) \leftarrow \neg p(t),\ \neg ab(t,t{+}1) \end{array}\ \}$$
$$P_{\neg ab} = \{\ \neg ab(t,t{+}1) \leftarrow \text{not}\ ab(t,t{+}1)\ \}$$

But that isn't right. $P_F \cup P_{\neg ab} \cup \mathit{Obs}$, for Obs a set of facts (ground literals) is equivalent to the default theory $(D'_{\neg ab}, \mathit{Obs})$ where

$$D'_{\neg ab} = \{\ \frac{p(t) \wedge \neg ab(t,t{+}1) :}{p(t{+}1)},\ \frac{\neg p(t) \wedge \neg ab(t,t{+}1) :}{\neg p(t{+}1)},\ \frac{:\, \neg ab(t,t{+}1)}{\neg ab(t,t{+}1)}\ \}$$

Here, the material implications F in $F \cup \mathit{Obs}$ have been replaced in $D_{\neg ab}$ by (monotonic, non-default) *rules*. That is not the same thing. In general, the default theory $(D \cup R,\ W)$ where R is a set of monotonic (non-default) rules is *not* equivalent to the default theory $(D,\ \mathit{mat}(R) \cup W)$.

Just to see the difference in this example, compute the answer set(s) of $P_F \cup P_{\neg ab} \cup \{\, p(1), \neg p(3) \,\}$. Don't do this by hand: use `clingo`. The clauses $P_F \cup P_{\neg ab}$ are in file `kautz_bad_reiter.lp`. What do you get?

`clingo` output is not always easy to read. You will need to put the output in a file and then edit it slightly to make it readable. Just submit the (edited) `clingo` output and add a brief commentary to explain the significance of the answers.

Question 4 Here is a common variation of the treatment of default persistence/inertia of the previous question. Some authors prefer to use the following frame axioms:

$$F' = \{\ \neg ab(t,t{+}1) \rightarrow (\, p(t) \leftrightarrow p(t{+}1)\,)\ \} \qquad D_{\neg ab} = \{\ \frac{:\, \neg ab(t,t{+}1)}{\neg ab(t,t{+}1)}\ \}$$

Note first that F' and F of Question 3 are logically equivalent. And moreover, F and F' are also logically equivalent to the conjunction of

$$\begin{gathered}
\neg ab(t,t{+}1) \rightarrow (\, p(t) \rightarrow p(t{+}1)\,) \\
\neg ab(t,t{+}1) \rightarrow (\, \neg p(t) \rightarrow \neg p(t{+}1)\,) \\
\neg ab(t,t{+}1) \rightarrow (\, p(t{+}1) \rightarrow p(t)\,) \\
\neg ab(t,t{+}1) \rightarrow (\, \neg p(t{+}1) \rightarrow \neg p(t)\,)
\end{gathered}$$

So (very rarely noticed!!) we have both 'forward' and 'backward' persistence. Amazingly, F' is often adopted as the natural frame axiom by authors who deny *vehemently* that 'backwards persistence' makes any sense.

Anyway. Put that to one side. Suppose we try to get the effects of the contrapositive[1] forms of F' by taking the following extended logic program P'_F:

$$\begin{gathered}
p(t{+}1) \ \leftarrow\ p(t),\ \neg ab(t,t{+}1) \\
\neg p(t{+}1) \ \leftarrow\ \neg p(t),\ \neg ab(t,t{+}1) \\
p(t) \ \leftarrow\ p(t{+}1),\ \neg ab(t,t{+}1) \\
\neg p(t) \ \leftarrow\ \neg p(t{+}1),\ \neg ab(t,t{+}1) \\
ab(t,t{+}1) \ \leftarrow\ p(t),\ \neg p(t{+}1) \\
ab(t,t{+}1) \ \leftarrow\ \neg p(t),\ p(t{+}1)
\end{gathered}$$

together with $P_{\neg ab} = \{\ \neg ab(t,t{+}1) \leftarrow \text{not } ab(t,t{+}1)\ \}$ as in the previous question.

Compute the answer sets of $P'_F \cup P_{\neg ab} \cup \mathit{Obs}$ for the following sets of observations Obs:

$$\{\, p(1),\ \neg p(3)\,\} \quad \text{and} \quad \{\, p(1),\ p(3)\,\} \quad \text{and} \quad \{\, p(2)\,\}$$

There are many ways of doing this. You can use `clingo`. The clauses $P'_F \cup P_{\neg ab}$ are in file `kautz_reiter.lp`. As usual, edit the output to make it more readable and add a little commentary.

[1] The 'contrapositive' of $P \rightarrow Q$ is $\neg Q \rightarrow \neg P$.

(If you want to try other values of 'maximum time' use `--const maxT=`N.)

Final remarks We haven't shown that the rules/clauses $P'_F \cup P_{\neg ab} \cup \mathit{Obs}$ are *equivalent* to the default theory $(D_{\neg ab},\ F' \cup \mathit{Obs})$. One could go on to investigate this (but we won't).

There are also other ways of formulating default persistence ('inertia') using Reiter default logic. The above is the most common.

Question 5 Here's another way of looking at the Stolen Car and similar problems. Let's take simple 'forward persistence' of $p(t)$ and $\neg p(t)$, i.e. P_{f} as in Question 1.

Now, if we have $p(t)$ and $\neg p(t{+}1)$ it must be because something happened in the transition from t to $t{+}1$: the car was stolen between t and $t{+}1$, or (being made of ice) it melted between t and $t{+}1$, or whatever. Let $q(t,t{+}1)$ represent that there is some car-disappearing action between t and $t{+}1$. We add the following 'effect axioms' to P_{f}:

$$\neg p(t{+}1) \leftarrow q(t,t{+}1) \qquad (\textit{'effects'})$$

Notice the deliberate asymmetry: we are allowing for actions that result in a transition from $p(t)$ to $\neg p(t{+}1)$ but we do not have actions that result in a transition from $\neg p(t)$ to $p(t{+}1)$. We could allow for such actions if we wanted to — we would add a clause $p(t{+}1) \leftarrow q(t,t{+}1)$. Or we could add a clause $p(t{+}1) \leftarrow q'(t,t{+}1)$ if we wanted to distinguish between actions that remove the car from the car park (q) and actions that put the car into the car park (q'). I won't do this. I want to make the point that we can have asymmetry between $p(t)$ and $\neg p(t)$.

So now: an action $q(t,t{+}1)$ happens, or it does not. We can represent that like this:

$$\begin{gathered}
q(t,t{+}1) \ \leftarrow \text{not } \neg q(t,t{+}1) \qquad (\textit{'exog'}) \\
\neg q(t,t{+}1) \ \leftarrow \text{not } q(t,t{+}1)
\end{gathered}$$

One last point. We need an 'action pre-condition': $q(t,t{+}1)$ could not happen if $p(t)$ is false. How do we express this pre-condition? I've pointed it out before. If b is an atom not appearing anywhere else in a program P, then

$$b \ \leftarrow\ L_1, \ldots, L_m,\ \text{not } b$$

added to P eliminates all answer sets containing $\{\, L_1, \ldots, L_m \,\}$. This is easy to see. Because b appears nowhere in P we can 'split' the program into the 'top part' $\{\, b \ \leftarrow L_1, \ldots, L_m,\ \text{not } b \,\}$ and the 'bottom part' P. If X is an answer set of P containing $\{\, L_1, \ldots, L_m \,\}$, simplifying the 'top part' leaves $\{\, b \leftarrow \text{not } b \,\}$, and this has no answer sets.

So to express the action pre-condition we can add the following:

$$b \ \leftarrow\ q(t,t{+}1),\ \neg p(t),\ \text{not } b(t) \qquad (\textit{'pre-conditions'})$$

Most ASP solvers, including `clingo` allow constraints to be written in the following form (but with `:-` for the arrow):

$$\leftarrow\ q(t,t{+}1),\ \neg p(t)$$

And finally, to guard against the possibility of answer sets with complementary literals, let us add the constraints:

$$\leftarrow\ p(t),\ \neg p(t) \qquad (\textit{'consistency'})$$

Note that `clingo` adds the 'consistency' constraints automatically.

So let P_{cars} be P_{f} together with the relevant instances of *'effects'*, *'exog'*, *'pre-conditions'*, and *'consistency'*.

(a) Check that the following are both answer sets of $P_{\text{cars}} \cup \{\, p(1), \neg p(3) \,\}$. You can easily do this by hand. (But do not submit it.) You could do it using splitting sets, but there is *no need*: you are only *checking* the stability condition for a *given* answer set. Just compute the reducts.

$$\{\, p(1), q(1,2), \neg p(2), \neg q(2,3), \neg p(3) \,\} \qquad \text{and} \qquad \{\, p(1), \neg q(1,2), p(2), q(2,3), \neg p(3) \,\}$$

This is as expected: one answer set explains the observation $\neg p(3)$ because it has the transition $q(1,2)$ and the other because it has the transition $q(2,3)$.

But perhaps more surprising is that the following is *also* an answer set of $P_{\text{cars}} \cup \{\, p(1), \neg p(3) \,\}$.

$$\{\, p(1), \neg q(1,2), p(2), \neg q(2,3), \neg p(3) \,\}$$

Check this. (Do not submit it.) Here, the car seems to disappear mysteriously.

Is this surprising? Well, the program as formulated says that an action $q(t, t{+}1)$ has the result that $\neg p(t{+}1)$. It does not say that absence of $q(t, t{+}1)$, i.e., $\neg q(t, t{+}1)$, cannot also have the effect $\neg p(t{+}1)$ and thereby override default persistence. You can see this already with the simpler example $P_{\text{cars}} \cup \{\, p(1), \neg p(2) \,\}$.

I recommend that you try these hand calculations but *do not* submit them. Instead submit your answer to the next part, part (b).

(b) Compute the answer sets of $P_{\text{cars}} \cup \{\, p(1), \neg p(2) \,\}$ and $P_{\text{cars}} \cup \{\, p(1), \neg p(3) \,\}$ using `clingo`. The clauses for P_{cars} are in file `kautz_with_q.lp`. Edit the output and add a *brief* comment.

(c) The answer sets of $P_{\text{cars}} \cup \{\, p(1) \,\}$ represent all the possible evolutions of the world from an initial state in which the car is in the car park.

Compute the answer sets of $P_{\text{cars}} \cup \{\, p(1) \,\}$ using `clingo`.

Which of these answer sets contain $\neg p(2)$?

You can do this by eye, or by adding the constraint

```
:- not -p(2).
```

(or more generally `:- not -p(maxT).`).

What is going on? In part (b) we calculate the answer sets of $P_{\text{cars}} \cup \{\, p(1) \,\}$ in which $\neg p(2)$ is true. That is *not* the same as calculating the answer sets of $P_{\text{cars}} \cup \{\, p(1) \,\} \cup \{\, \neg p(2) \,\}$!

In general: if we want to determine how some set of observations *Obs* ($p(t)$ and $\neg p(t)$ literals) could have come about, we do not compute answer sets of $P_{\text{cars}} \cup \mathit{Obs}$. No: we calculate the answer sets of $P_{\text{cars}} \cup \{\, p(1) \,\}$. These are the possible evolutions of the car park world from a $p(1)$ state. Then we pick out just those answer sets (possible evolutions)

which have *Obs* all true. (Cf. the action language $\mathcal{C}+$ covered at the end of the course.) Or, if we are interested in cases where initially the car is not in the car park, we calculate the answer sets of $P_{\text{cars}} \cup \{\, \neg p(1) \,\}$ and then check which have *Obs* all true.

Try this for the case where, say, maximum time `maxT = 6`, and we have observations $p(1)$ and $\neg p(6)$. Submit a (suitably edited) copy of the `clingo` output.

The Hanks-McDermott Problem (Yale Shooting Problem)

Now we are going to do a similar exercise for an even more famous problem in the AI literature: the Yale Shooting Problem, also known as the Hanks-McDermott Problem.

Hanks, S. and McDermott, D. Nonmonotonic logic and temporal projection. *Artificial Intelligence* 33:379–412 (1987).

A *hugely* influential paper (for better or worse, depending on your point of view).

The problem arises with the formalisation of a simple example concerning the effects of loading and shooting a gun, and letting time pass (waiting). A person is initially alive. Shooting someone with a loaded gun results in their death. The action *wait* has no effect on truth of fluents. The expected model (or extension if we use Reiter default logic) is thus one where the victim ends up dead after the sequence *load*; *wait*; *shoot*.

The problem? A natural (or perhaps simple-minded) formalisation of default persistence ('inertia') gives two *different* models!!

- one (as expected) in which the victim is not alive after the sequence *load*; *wait*; *shoot*;
- and another (unexpected) model in which the victim is alive, but the gun has somehow become unloaded during the *wait*. This second, unexpected model is sometimes called the 'anomalous' model.

Much of the force of the Hanks-McDermott paper is that the problem is not specific to any particular default reasoning formalism: they show in their paper that the same problem arises using several non-monotonic formalisms. (That part is not at all surprising now, since the relationships between these formalisms are well understood.)

Hanks & McDermott's conclusion: this appears to be a problem for default and/or temporal reasoning *in general*, rather than just an objection to a particular proposal.

The problem attracted the attention of a great many researchers and huge numbers of papers were produced proposing a variety of solutions. Responses ranged from the view that the problem is not really a problem, to the opinion that it constitutes a fatal objection to the whole project of logic-based AI itself. There is also a range of solutions, and demonstrations that other temporal reasoning systems are immune to the problem.

Question 6 Here is a formulation in Reiter default logic similar to the one we had earlier. The only difference is that we now have two fluents that persist, *alive* and *loaded* (and their negations).

Defaults:

$$\{ \ \frac{: \neg ab(loaded, t, t{+}1)}{\neg ab(loaded, t, t{+}1)}, \ \frac{: \neg ab(alive, t, t{+}1)}{\neg ab(alive, t, t{+}1)} \ \}$$

Persistence/frame (as before):

$$\begin{array}{l} \{ \ \neg ab(loaded, t, t{+}1) \rightarrow (\ loaded(t) \leftrightarrow loaded(t{+}1) \ ), \\ \quad \neg ab(alive, t, t{+}1) \rightarrow (\ alive(t) \leftrightarrow alive(t{+}1) \ ) \ \} \end{array}$$

And we need to specify the effects of the *load* and *shoot* actions:

$$\begin{array}{l} \{ \ load(t) \rightarrow loaded(t{+}1), \\ \quad loaded(t) \wedge shoot(t) \rightarrow \neg alive(t{+}1) \ \} \end{array}$$

This is not exactly the same formalisation as that of Hanks-McDermott — they use the 'situation calculus' — but these differences are immaterial. The treatment of default persistence and all essential features of the formalisation are the same.

We want to compute the extension(s) of this default theory when we add the further facts $\{ \ load(0), wait(1), shoot(2) \ \}$.

Note that times range here from 0 to 3, rather than from 1 to 4. (There is no particular reason why I chose that. It does not affect anything.) Also, the occurrences of actions are represented in the form $shoot(2)$ and not in the form $shoot(2, 3)$ as was used in the 'Stolen Car'. That does not affect anything. And note finally that here shooting does not make the gun unloaded. We could change that but nothing relevant to the Hanks-McDermott problem is affected (try it).

Now, we want to compute the extensions. We can do this exactly as for the 'Stolen Car' examples. Let's translate to a logic program and use `clingo`. The `clingo` clauses are in files `ysp_reiter.lp` and `ysp_effects.lp`. Notice that `ysp_effects.lp` has two clauses corresponding to the effects axiom $loaded(t) \wedge shoot(t) \rightarrow \neg alive(t{+}1)$, otherwise we do not get the full effect of the material implication $\rightarrow$.

So: what are the extensions (answer sets)?

(Hanks & McDermott say in their paper that there are just two. They are wrong.)

How are these extensions to be interpreted? (*Briefly.*)

(*Optional* Do not submit this, but you could investigate what happens if you remove the contrapositive half of the effects of *shoot*. Or if *shoot* also unloads the gun.)

Question 7 It is generally held that the essence of the Hanks-McDermott problem is the existence of the 'anomalous model', which in itself has nothing to do with loading the gun, or shooting, or anything else.

Many subsequent papers discuss the problem in the following simplified form:

$$\begin{array}{l} \{ \ loaded(1), \\ \quad alive(2), \\ \quad loaded(2) \rightarrow \neg alive(3) \ \} \end{array}$$

together with the usual default rules for persistence ('inertia').

(Notice that the last formula is logically equivalent to $\neg loaded(2) \vee \neg alive(3)$.)

What are the extensions for this simplified version of the problem? The `clingo` clauses are in file `ysp_simplified.lp`. How do you interpret the answer?

Question 8 (Follow up: Recommended but *unassessed*. Do not submit!!)

What do we get if instead of the Reiter-like default persistence rules tried by Hanks and McDermott we use instead only the simple 'forward persistence' rules we looked at for the 'Stolen Car' in Questions 1 and 5? That didn't deal with the 'Stolen Car' very well but what happens here?

The `clingo` clauses are in file `ysp_forward.lp`. What are the answer sets?

You should find that the 'anomalous model' has disappeared. Surprised?

One common response to the Hanks-McDermott paper is that it is not at all surprising or undesirable that the gun might mysteriously unload itself during the waiting. We might be waiting for a very long time before shooting, and there is nothing in the example as formalised to say this could not happen. Why couldn't the gun become unloaded during the wait? (What is not clear to me is why this argument does not apply equally to being alive. Why then could the victim not die during the waiting — or indeed during the loading or during the shooting?)

The 'simple forward' persistence formulation eliminates all these possibilities. But suppose that we change the example slightly. Suppose now that *wait* might unload the gun, or it might not. How could the 'simple forward' persistence formulation of `ysp_forward.lp` be modified to allow for this? In other words, suppose we *do* want two models/extensions here: one in which the gun does not unload during the wait (and so the shooting kills), and the other in which the gun does unload during the wait (and so the shooting does not kill). How do we get this, with 'simple forward' persistence? We don't want the victim dying, except by being shot with a loaded gun.

There are two ways to do it. Do *both*. (Optional. Don't submit!!) Here they are:

Method (1) Proceed as for the 'Stolen Car' in Question 5. Introduce an action — call it *unloading* — which unloads the gun, together with the appropriate pre-conditions, including a pre-condition that says *unloading* can only happen during *wait*.

Method (2) Do not bother with *unloading*. Think of a way of expressing in clauses that *wait*(T) might cause $\neg$*loaded*$(T+1)$ or might not. One way to think of it is this: if *wait*(T) happens then either *loaded*$(T+1)$ or $\neg$*loaded*$(T+1)$. Or look at how *may cause* works in the action language $\mathcal{C}+$ covered in the last part of the course. But be careful: *wait*(T) could unload a loaded gun but we don't want to say that *wait*(T) could load a gun that is not loaded.

There is nothing to submit for this part. I recommend that you try it however, and run your solutions in `clingo`. Or at the very least look at the solution I will provide in my sample answer.

SUBMISSION: Summary

Q0. Nothing to submit. But read it.

Q1. Use 'splitting sets' to determine the answer sets of $P_{\rm f} \cup Obs$ for the following sets of observations *Obs*:

$$\{\, p(1),\ \neg p(3)\,\} \quad \text{and} \quad \{\, p(1),\ p(3)\,\} \quad \text{and} \quad \{\, p(2)\,\}$$

Q2. Use 'splitting sets' to determine the answer sets of $P_{\rm f} \cup P_{\rm b} \cup Obs$ for the following sets of observations *Obs*:

$$\{\, p(1),\ \neg p(3)\,\} \quad \text{and} \quad \{\, p(1),\ p(3)\,\} \quad \text{and} \quad \{\, p(2)\,\}$$

Q3.

(a) Show that the following are both extensions of $(D_{\neg ab},\ F \cup \{\, p(1), \neg p(3)\,\})$.

$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3)\,\} \cup \{\, \neg ab(1,2), ab(2,3)\,\})$$
$$\mathrm{Th}(F \cup \{\, p(1), \neg p(3)\,\} \cup \{\, ab(1,2), \neg ab(2,3)\,\})$$

Part (a) you do by hand. (You write something.)

(b) Nothing to submit. (But I strongly recommend that you do the exercise, or at least look at the sample solution when it is published.)

(c) Use `clingo` to compute the answer set(s) of $P_F \cup P_{\neg ab} \cup \{\, p(1), \neg p(3)\,\}$.

Submit (edited) copies of the `clingo` outputs, with a few comments.

Q4. Use `clingo` to compute the answer sets of $P'_F \cup P_{\neg ab} \cup Obs$ for the following sets of observations *Obs*:

$$\{\, p(1),\ \neg p(3)\,\} \quad \text{and} \quad \{\, p(1),\ p(3)\,\} \quad \text{and} \quad \{\, p(2)\,\}$$

Submit (edited) copies of the `clingo` outputs, with a few comments.

Q5.

(a) Nothing to submit. (But do the calculations.)

(b) Use `clingo` to compute the answer sets of $P_{\rm cars} \cup \{\, p(1), \neg p(2)\,\}$ and $P_{\rm cars} \cup \{\, p(1), \neg p(3)\,\}$. Comment briefly.

(c) Use `clingo` to compute the answer sets of $P_{\rm cars} \cup \{\, p(1)\,\}$ for various values of `maxT`.

For `maxT` $= 6$, which of these answer sets contain $\neg p(6)$? Does this make sense?

Q6. Use `clingo` to compute the extensions of the Hanks-McDermott default logic formalisation of the YSP. Comment briefly on what you find.

Q7. Use `clingo` to compute the extensions of the simplified version of the YSP. Comment briefly.

Q8. Optional follow up: recommended but unassessed. Nothing to submit.

It is *NOT* necessary to type submissions. Handwritten submissions are perfectly acceptable, as long as they are readable. If you do type your submission, please do not submit it electronically.

Where it says 'add brief comments to the `clingo` output' it means brief. The comments are just for me to check you have understood the significance, for the example, of the answers `clingo` computes.

FEEDBACK

This exercise is intended to provide a directed exploration of issues in the formalisation of default persistence ('inertia') and temporal reasoning, and thereby to complement the material presented in lectures. It is also intended to provide a framework for **revision** and practising the core material on answer sets and default logic. There is an assessment component but as you can see, it is not the primary purpose. I have tried to keep the amount of writing to a minimum.

Submissions will be marked and returned with comments in the usual way. For revision purposes I will also publish a sample solution, with commentary, on the course website.

The main aim of this exercise is the directed exploration. The assessment component is secondary. If you have questions about the sample answer, or about the coursework, or about anything else in the course, **please ask**.