Assignment Four: Logic Programming Introduction

Due: 11:59pm, Thursday 7 March 2024. Submit solution online using Canvas

These questions should be done with SWI Prolog. See the web page for how to download, install and run SWI Prolog.

Submit your answers using Canvas. The file for question 2 must run in Prolog (any text answers should be in comments). Use proper sentences in your answers. Make sure you name(s), student number(s) is at the top of each file. In your submission, use only non-proprietary formats (e.g., text, pdf, jpeg but not Word or Excel or Pages). You may work is a group of size one or two. Ask questions on Piazza.

Question One

Consider the facts in `http://cs.ubc.ca/~poole/cs312/2024/as4/cs312_2024.pl`.

The queries that you provide should work for any knowledge base with the same predicates. You are not allowed to use any built-in predicates.

For each question (a) to (d):

i) Give a (conjunctive) query to answer the question or explain why there cannot be a conjunctive query. (Prolog's answer may provide more information than is asked for, but it should not be wrong.)

ii) If the query is possible, give *all* of the answers from the facts in `cs312_2024.pl`.

iii) If there are no answers, suggest some fact(s) that could be added so that the answer is not "false", and give the answer for the query given the new facts. Make sure you test it!

(a) "What dates in March are assignments due?"

(b) "What is the email of a TA of cs312 in 2024?" [Note that not all of the emails are for TAs.]

(c) "what day of the week has office hours by different people?"

(d) "What are the first names of people that have office hours on consecutive days?"

(e) "Is there a TA who is not holding office hours?"

(f) What information would be required to compute *number_of_classes_before* from *exam*? (You do not need to write any Prolog for this. Hint: try to compute it yourself and take notice of any information you required.)

Question Two

Consider the domain of house plumbing represented in Figure 1. In this figure, p_1, p_2 and p_3 are water pipes. p_1 is the pipe coming in from the main water supply. t_1, t_2 and t_3 are taps and d_1, d_2 and d_3 are drainage pipes. The other labels should be obvious.

Suppose we have the following atoms:

- *pressurized_*p_i, is true if pipe p_i has mains pressure in it. Pipe p_1 is always pressurized. Other pipes are pressurized if they are connected to a pressurized pipe through an open tap.
- *on_*t_i is true if tap t_i is on.

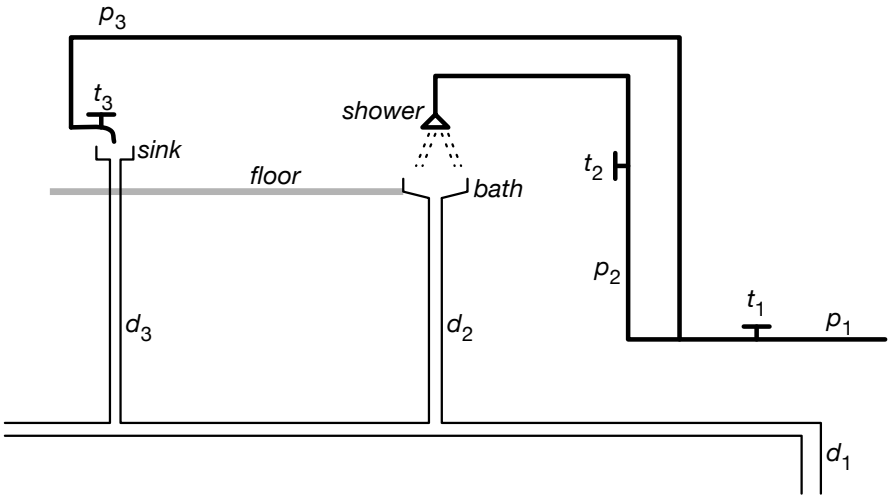

Figure 1: The Plumbing Domain

- *off*_t_i is true if tap t_i is off.
- *wet*_b is true if b is wet.
- *flow*_c is true if water is flowing through component c.
- *plugged*_c is true if component c has the plug in.
- *unplugged*_c is true if component c doesn't have the plug in.

Assume the taps and plugs have been in the same positions for one hour; you don't need to consider the dynamics of turning on taps and inserting and removing plugs.

The file `http://www.cs.ubc.ca/~poole/cs312/2024/as4/plumbing.pl` contains a Prolog axiomatization for how water can flow down drain $d1$ if taps t_1 and t_2 are on and the bath is unplugged.

(a) Finish the axiomatization for the sink in the same manner as the axiomatization for the bath. Give the intended interpretation (using Prolog comments) for all atoms you introduce.

Test it with various configurations of taps on and off, and the plug in and out.

(b) Axiomatize how the floor is wet if the sink overflows or the bath overflows. Each will overflow if the plug is in and water is flowing in. You may invent new atoms as long as you give their intended interpretation.

(c) Suppose a hot water system is installed to the left of tap t_1. There is another tap in the pipe leading into the how water system, and it supplies hot water to the shower and the sink (there are separate hot and cold water taps for each). Add this to your axiomatization. You don't need to model water temperature. Give the intended interpretation for all atoms you invent.

You need to hand in a complete listing of your program (including the intended interpretation for all symbols) and a trace of a session to show it runs for various setting of taps on and off, and plugs in and out.

Question Three

Given the Prolog program in
`http://www.cs.ubc.ca/~poole/cs312/2024/as4/sillyeg.pl`

(a) Show how the bottom-up proof procedure works for this example. In particular show a sequence of atoms added to the consequence set together with the corresponding clauses selected.

(b) Give a failing derivation for query: `?- slithy.`

(c) Give a succeeding derivation for query: `?- slithy.`

(d) Give the box model for `slithy`. Use this model to explain a trace of the Prolog's proof. In particular, explain one instance of each of $\{call, exit, redo, fail\}$ when you use Prolog's "trace." command, for the query `?- slithy.`

(e) How many answers does Prolog give for this query? Explain why there are this many answers.

Question Four

For each question, specify how long you spend on it, and what you learned. Was the question reasonable? (This question is part of the assignment, so please do it!)