Assignment Five: Logic
Due: 11:59 pm, Tuesday 13 February 2018.

Solving following problems requires using ailog (http://artint.info/code/ailog/) and/or the Python code at http://aipython.org/aipython_322_as5.zip (now includes propositional logic). To use ailog you need to install SWI Prolog.

This can be done in groups of size 1, 2 or 3. Working alone is not recommended. All members of the group need to be able to explain the group’s answer.

Submit your answers using Canvas. Use proper sentences (not note form) in your answer. Ask questions on Canvas discussion board. Feel free to answer them too.

Question One

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:
- \( \text{pressurised}_{p_i} \) is true if pipe \( p_i \) has mains pressure in it. \( p_1 \) is always pressurized. Other pipes are pressurized if they are connected to a pressurized pipe through an open tap.
- \( \text{on}_{t_i} \) is true if tap \( t_i \) is on.
- \( \text{off}_{t_i} \) is true if tap \( t_i \) is off.
- \( \text{wet}_b \) is true if \( b \) is wet.
- \( \text{flow}_c \) is true if water is flowing through component \( c \).
- \( \text{plugged}_c \) is true if component \( c \) has the plug in.
- \( \text{unplugged}_c \) 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.

You can either use AILog or the definite clause interpreter in Python. The file http://www.cs.ubc.ca/~poole/cs322/2018/as5/plumbing.ailog contains a AILog axiomatization for how water can flow down drain \( d_1 \) if taps \( t_1 \) and \( t_2 \) are on and the bath is unplugged. http://www.cs.ubc.ca/~poole/cs322/2018/as5/plumbing.py contains a Python representation (and is also in the Python distribution).

(a) Show how the bottom-up proof procedure for computing consequences of KB works for this example. For every atom added to the consequence set, give the clause that was used to infer it.

(b) Give a top-down derivation for the query \( \text{flow}_{d_1} \).

(c) Finish the axiomatization for the sink in the same manner as the axiomatization for the bath.

(d) What information would you expect the occupant of the house to be able to provide that the plumber can’t? Change the axiomatization so that such information is asked of the user of the system.

(e) Axiomatize how the floor is wet if the sink overflows or the bath overflows. They overflow if the plug is in and water is flowing in. You may invent new atoms as long as you give their intended interpretation. [Assume that the taps and plugs have been in the same positions for one hour; you don’t need to axiomatize the dynamics of the turning on taps and inserting and removing plugs.]
Figure 1: The Plumbing Domain

You need to hand in your program, including the intended interpretation for all symbols used, and a trace of the AILog or Python session or to show it runs.

**Question Two**

This question considers using integrity constraints and consistency-based diagnosis in an agent that interacts with various information sources on the web. To answer a question, the agent will ask a number of the information sources for facts. However, information sources are sometimes wrong. It is useful to be able to automatically determine which information sources may be wrong when a user gets conflicting information.

In this question explores how integrity constraints and assumables can be used to determine what errors are present in different information sources.

This question uses meaningless symbols such as $a$, $b$, $c$, . . ., but in a real domain there will be meaning associated with the symbols, such as $a$ meaning “there is skiing in Hawaii” and $z$ meaning “there is no skiing in Hawaii”, or $a$ meaning “butterflies do not eat anything” and $z$ meaning “butterflies eat nectar”. We will use meaningless symbols in this question because the computer does not have access to the meanings and must simply treat them as meaningless symbols.

Suppose the following information sources and associated information are provided:

**Source $s_1$:** Source $s_1$ claims the following clauses are true:

\[
\begin{align*}
a & \leftarrow h. \\
d & \leftarrow c.
\end{align*}
\]

**Source $s_2$:** Source $s_2$ claims the following clauses are true:

\[
\begin{align*}
e & \leftarrow d. \\
f & \leftarrow k. \\
z & \leftarrow g. \\
j.
\end{align*}
\]

**Source $s_3$:** Source $s_3$ claims the following clause is true:

\[
\begin{align*}
h & \leftarrow d.
\end{align*}
\]
Source $s_4$: Source $s_4$ claims the following clauses are true:

\[ a \leftarrow b \land e. \]
\[ b \leftarrow c. \]

Source $s_5$: Source $s_5$ claims the following clause is true:

\[ g \leftarrow f \land j. \]

Yourself: Suppose that you know that the following clauses are true:

\[ \text{false} \leftarrow a \land z. \]
\[ c. \]
\[ k. \]

Not every source can be believed, because together they produce a contradiction.

(a) Code the knowledge provided by the users into Python representation or AILog using assumables. To use a clause provided by one of the sources, you must assume that the source is reliable.

(b) Use the program to find all of the conflicts about what sources are reliable. (To find conflicts in AILog, ask \texttt{false}, and then use \texttt{more}.)

(c) Which single source(s), if unreliable, could account for a contradiction (assuming all other sources were reliable)? Explain how you know this, and explain the significance.

(d) What are the minimal diagnoses? Explain how you got these, and explain the significance of them.

Question Three

For each question, specify how long you spend on it, and what you learned. How was the work in the team allocated? Was the question reasonable? (This question is worth marks, so please do it!)