G53KRR 2007-8 model answers

Question 1 **(a)** A representation language is a formal language used to represent information. Reasoning or inference mechanism helps to extract more information from what is explicitly represented. Possible examples include: first order logic as a representation language and resolution as an inference mechanism; production system rules as representation language, and forward chaining as an inference mechanism.

(b) The main problems are: computational intractability and inability to express default rules, probabilistic and approximate knowledge.

(c) An inference system is sound (with respect to some semantical notion of entailment) if it only derives sentences which are entailed by the premises. It is complete if it can derive all sentences which are entailed by the premises.

(d) Resolution is sound but not complete; for example it is impossible to derive $p \lor \neg p$ by resolution from an empty set of sentences. It is however complete for refutation: if an empty clause (a contradiction) is entailed by a set of sentences, then it is possible to derive it using resolution.

Question 2 **(a)** **S1** $\forall x(Plant(x) \land \neg Water(x) \supset \neg Survive(x))$

S2 $\neg \exists x Responsible(x, bamboo) \supset \neg Water(bamboo)$

S3 $\neg \exists x Responsible(x, bamboo) \supset \neg Survive(bamboo)$

To show that S1 and S2 do not logically entail S3, consider the following interpretation $M = (D, I)$: $D = \{b\}$, $I(bamboo) = \{b\}$, $I(Plant) = \{\}$, $I(Water) = \{\}$, $I(Survive) = \{b\}$, $I(Responsible) = \{\}$. In M, S1 is true because nothing is a plant, S2 is true because nothing is getting any water, and S3 is false because $\neg \exists x Responsible(x, bamboo)$ is true and $\neg Survive(bamboo)$ is false. Intuitively, in this interpretation bamboo is going to survive without water because it is not considered to be a plant.

(b) The missing information is that bamboo is a plant:

S4 $Plant(bamboo)$

To show that S1, S2, S4 entail S3, consider an arbitrary interpretation $M = (D, I)$ where S1, S2 and S4 are true. We need to show that S3 is also true in M. Suppose that $\neg \exists x Responsible(x, bamboo)$ is true in M (if it is false, we are done, because then S3 is trivially true). We need to show that $\neg Survive(bamboo)$ is true. $\neg \exists x Responsible(x, bamboo)$ being true means that $\neg Water(bamboo)$ is true because of S2. Since $Plant(bamboo)$ is true and $\neg Water(bamboo)$ is true, then because of S1 $\neg Survive(bamboo)$ is true, which is what we needed to show.

Question 3 **(a)** **C1** $[Sum(x, y, f(x, y))]$

C2 $[\neg Sum(x, y, z_1), \neg Sum(x, y, z_2), (z_1 = z_2)]$

C3 $[\neg Sum(x, y, z), Sum(y, x, z))]$

(b) We need to show that from C1, C2, C3 and negation of the goal we can derive an empty clause. The negation is $[\neg Happy(friendOf(joe))]$. Here is the derivation:

1 $[\neg Superman(x), \neg Helps(x, friendOf(joe))]$ by resolution from $[\neg Happy(friendOf(joe))]$ and C3, with substitution $y/friendOf(joe)$

2 $[\neg Superman(joe)]$ from 1 and C1, with substitution x/joe

3 [] from 2 and C1.

Question 4 **(a)** A Horn clause is a clause with at most one positive literal. A clause which contains a positive literal is a positive or definite clause. A clause which does not contain a positive literal, including an empty clause, is a negative clause.

(b) **input:** a finite set of atomic sentences $q_1, \ldots, q_n$

output: YES if KB entails all of q_i, NO otherwise

procedure: SOLVE$[q_1, \ldots, q_n]$

- **if** n = 0 **then return** YES
- **for** each clause $c \in KB$ **do**
  - **if** $c = [q_1, \neg p_1, \ldots, \neg p_m]$ and SOLVE $[p_1, \ldots, p_m, q_2, \ldots, q_n]$ **then return** YES
- **end for**
- **return** NO

(c) Goal ordering is used to make the process of reasoning more efficient (and sometimes to ensure termination). An example where the goal order matters is e.g.

$$\texttt{AmericanCousin}(\texttt{x}, \texttt{sally}) : -\texttt{American}(\texttt{x}), \texttt{Cousin}(\texttt{x}, \texttt{sally})$$

where to solve the goal we need to generate an American first and then check if he or she is a cousin of Sally; this may require going through all possible Americans and is inefficient. On the other hand,

$$\texttt{AmericanCousin}(\texttt{x}, \texttt{sally}) : -\texttt{Cousin}(\texttt{x}, \texttt{sally}), \texttt{American}(\texttt{x})$$

first generates a cousin of Sally; Sally's cousins are likely to be less numerous than Americans so this is more efficient.

(d) Negation as failure in Prolog: if fail to derive A, assert **not**A. Negation as failure in production systems: if failed to match A in the working memory, match **not**A.

Question 5 **(a)** Development cycle of a knowledge based system:

```
                   1. Plan knowledge base
                          |
                          v
                   2. Select domain experts
                   and knowledge sources
                          |
                          v
                   3. Elicit knowledge
                          |
                          v
                   4. Formulate and represent
                   knowledge
Continue                  |
knowledge                 v
acquisition        5. Implement knowledge base
                          |
    ^                     v
    |              6. Test knowledge base
    |                     |
    |        no           v
    -------------------- OK?
                          | yes
                          v
                   7. Systems test
```

In 1, the content of the knowledge base, relevant inputs and outputs, strategy for testing, knowledge dictionary, concepts etc. are identified.

In 2, sources of knowledge are selected (domain experts, corpora, etc.).

In 3, the process of knowledge elicitation takes place.

In 4, the knowledge is formulated in the form suitable for inference.

In 5, it is encoded in machine-readable form.

(b) First we need to decide on all conditions relevant for making a decision, and all possible actions. A decision table consists of a list of condition stubs (possible relevant conditions), action stubs (progress), condition entries (yes, no, and - for not relevant, for each condition) and action entries (which action is to be taken if specified conditions hold).

Let us assume that relevant conditions are: students's module average, number of passed credits, and number of hard fails.

```
                || Rule 1 | Rule 2 | Rule 3  | Rule 4   |
----------------||------- |--------|---------|----------|
```

```
Passed credits  || >=120  | >= 80  | >= 90   | >= 100   |
                ||        |        |         |          |
Module average  ||   -    | >= 40  | >= 45   | >= 50    |
                ||        |        |         |          |
Hard fails      ||   -    |  0     | <= 1    | <= 2     |
----------------||------- |--------|---------|----------|
----------------||------- |--------|---------|----------|
                ||        |        |         |          |
Progress        ||   X    |    X   |   X     |    X     |
                ||        |        |         |          |
```

The resulting system will have rules corresponding to columns, e.g. Rule 1 says that if passed credits are at least 120, then progress.

Question 6 **(a)** Nodes are boolean variables, arcs correspond to causal relationships. Each propositional variable in the belief network is conditionally independent from the nonparent variables given the parent variables. This is a more compact way of representing probabilistic information than joint probability distribution.

(b)

```
c
|
|
v
h -------> g
^
|
|
w
```

Example independence assumptions (where C is c or $\neg c$, similarly for other variables):

$$Pr(W|C) = Pr(W)$$

$$Pr(C|W) = Pr(C)$$

$$Pr(G|C \wedge W \wedge H) = Pr(G|H)$$

(c) By the chain rule,

$$Pr(C \wedge W \wedge H \wedge G) = Pr(C) \cdot Pr(W|C) \cdot Pr(H|C \wedge W) \cdot Pr(G|C \wedge W \wedge H)$$

By the independence assumptions in the network, we have

$$Pr(W|C) = Pr(W)$$

$$Pr(G|C \wedge W \wedge H) = Pr(G|H)$$

So

$$Pr(C \wedge W \wedge H \wedge G) = Pr(C) \cdot Pr(W) \cdot Pr(H|C \wedge W) \cdot Pr(G|H)$$

$Pr(g|c \wedge w) = \frac{Pr(g\wedge c\wedge w)}{Pr(c\wedge w)} = \frac{Pr(g\wedge c\wedge w\wedge h)+Pr(g\wedge c\wedge w\wedge\neg h)}{Pr(c)\cdot Pr(w)}$
$= \frac{Pr(c)\cdot Pr(w)\cdot Pr(h|c\wedge w)\cdot Pr(g|h)+Pr(c)\cdot Pr(w)\cdot Pr(\neg h|c\wedge w)\cdot Pr(g|\neg h)}{Pr(c)\cdot Pr(w)} =$
$\frac{Pr(c)\cdot Pr(w)\cdot (Pr(h|c\wedge w)\cdot Pr(g|h)+Pr(\neg h|c\wedge w)\cdot Pr(g|\neg h))}{Pr(c)\cdot Pr(w)}$
$= Pr(h|c\wedge w)\cdot Pr(g|h)+Pr(\neg h|c\wedge w)\cdot Pr(g|\neg h) = 0.9\cdot 0.6+0.1\cdot 0.1 =$
$0.54 + 0.01 = 0.55$ where we use $Pr(\neg h|c \wedge w) = 1 - Pr(h|c \wedge w)$.