Lab 6

Propositional Logic: Models

Learning objectives for this week are:

1. To understand how to encode knowledge in a form suitable for Prover9 and Mace4
2. To see how obtaining models for a given set of clauses is connected with building proofs

Henceforth, we will use for our experiments a knowledge base named KB, comprising one or more clauses connected by a logical AND. We will focus on testing whether KB has at least one model (a set of assignments of truth values to its propositional variables which make it true) and whether a given goal sentence g could be proven based on the KB content. We will say a proposition is $satisfiable$ if it has at least one model; otherwise, it is $unsatisfiable$. A proposition is *valid* if it is true no matter the truth values assigned to its components.

6.1 Formalizing puzzles: Princesses and tigers

We will start with formalizing a logical puzzle described in [1], who borrowed it from [2].

A prisoner is given by the King holding him the opportunity to improve his situation if he solves a puzzle. He is told there are three rooms in the castle: one room contains a lady and the other two contain a tiger each. If the prisoner opens the door to the room containing the lady, he will marry her and get a pardon. If he opens a door to a tiger room though, he will be eaten alive. Of course the prisoner wants to get married and be set free than being eaten alive.

The door of each room has a sign bearing a statement that may be either true or false. The sign on the door of the room containing the lady is true and at least one of the signs on the doors of the rooms containing tigers is false. The signs say respectively:

1. There is a tiger in room #2.
2. There is a tiger in this room.
3. There is a tiger in room #1.

Exercise 6.1 *Do you know in which room the lady is?*

Let's try to formalize what we know as an input for Prover9/Mace4, except won't specify the goal we intend to prove (a.k.a. the conjecture). Once done this, we will use Mace4 to see how many models it can find. If we obtain just one model (which is usually the case when dealing with this type of puzzles), we can see the solution immediately; for example, we will know the lady is in room #1. Then, we can add this finding as a goal to the input file and use Prover9 to build a proof for it. If we end up with more than one model, there is still hope: either in all of them the lady is in the same room, or we can ask for more clues.

Exercise 6.2 *How can we express the fact that there exists a lady in the castle? Write this information alone in the* `oneLadyTwoTigers-step1.in` *file, then call Mace4 to generate as many models as it can for this tiny KB:* `mace4 -c -n 2 -m -1 -f oneLadyTwoTigers-step1.in | interpformat`*. How many models do you get? How come?*

Now, we will tell the system the lady is unique (i.e., there is only one lady in the castle). The knowledge base looks like this:

Listing 6.1: Step 2: There is precisely one lady in the castle.

```
assign(max_seconds, 30).
set(binary_resolution).

formulas(assumptions).
 %there is a lady in room 1, 2 or 3
 l1 | l2 | l3.
 %no lady in more than 1 room; the princess is unique
 l1 -> -l2.
 l1 -> -l3.
 l2 -> -l1.
 l2 -> -l3.
 l3 -> -l1.
 l3 -> -l2.
end_of_list.

formulas(goals).
end_of_list.
```

Exercise 6.3 *How many models for the new KB does Mace4 produce now?*

We go on by specifying that there are 2 tigers in the castle, in separate rooms, and there is no tiger in the room where the lady stays.

Listing 6.2: The lady and the tigers stay in separate rooms.

```
assign(max_seconds, 30).
set(binary_resolution).

formulas(assumptions).
 %there is a lady in room 1, 2 or 3
 l1 | l2 | l3.
 %no lady in more than 1 room; the princess is unique
 l1 -> -l2.
 l1 -> -l3.
 l2 -> -l1.
 l2 -> -l3.
 l3 -> -l1.
 l3 -> -l2.

 %there are 2 tigers in two of the rooms 1, 2 or 3
 t1 & t2 | t2 & t3 | t1 & t3.

 %no tiger in the room where the lady stays
 l1 -> -t1.
```

```
 l2 -> -t2.
 l3 -> -t3.
 %do we also have to write t1 -> -l1 and alike, or do we already have that?

end_of_list.

formulas(goals).
end_of_list.
```

Exercise 6.4 *We wrote* `l1 - > -t1` *in order to say "if the lady is in room #1, then there is no tiger in the first room". Do we also have to write* `t1 - > -l1` *in order to say "tiger in room #1 means no lady in room #1"? Why (not)?*

So far, we have formalized the common sense knowledge about the field. Although obvious, this knowledge is vital for the systems to be able to build models and to exclude some of those inconsistent with the problem statement. Further, we will encode the clues on the doors. This part requires a certain effort from the programmer as in Propositional Logic we are not allowed to use sentences *about some other sentences*: we can't simply say "if John is from country C, then, for every sentence P John says, $\neg P$ is true".

Thus, we will need to interpret the meaning of the sentence on each door and tell Prover9 the result of this interpretation. As an example, if we assume the princess is in room #3, then the clue on the door is true, so we jump to the conclusion a tiger is present in room #1. What we will state is precisely this: `l3 -> t1`. The full input file is:

Listing 6.3: One lady and two the tigers: full KB.

```
assign(max_seconds, 30).
set(binary_resolution).

formulas(assumptions).
 %there is a lady in room 1, 2 or 3
 l1 | l2 | l3.
 %no lady in more than 1 room; the princess is unique
 l1 -> -l2.
 l1 -> -l3.
 l2 -> -l1.
 l2 -> -l3.
 l3 -> -l1.
 l3 -> -l2.

 %there are 2 tigers in two of the rooms 1, 2 or 3
 t1 & t2 | t2 & t3 | t1 & t3.

 %no tiger in the room where the lady stays
 l1 -> -t1.
 l2 -> -t2.
 l3 -> -t3.


 %clue on door #1: there is a tiger in room #2
 l1 -> t2.

 %clue on door #2: there is a tiger here
 l2 -> t2.

 %clue on door #3: there is a tiger in room #1
 l3 -> t1.

```

```
 %at least one of the clues on tiger rooms lies.
 (t1 & t2) -> (-t2 | -t2).
 (t2 & t3) -> (-t2 | -t1).
 (t1 & t3) -> (-t2 | -t1).

end_of_list.

formulas(goals).

end_of_list.
```

Exercise 6.5 *Ask Mace4 to generate all models. How many do you get? Is the lady in the same room in all of them? If so, add the position of the lady to the input file as a conjecture and ask Prover9 to prove it. Take a look at the proof.*

Exercise 6.6 *Let's drop the assumption that at least one of the clues on the tiger room lies. Ask Mace4 to generate all models in this situation. How many do you get? Is the lady in the same room in all of them?*

Let us assume we have one room containing the lady, one containing a tiger and one empty room. If the prisoner opens the door to the room containing the lady, he will marry her and get a pardon. If he opens a door to a tiger room though, he will be eaten alive. If he opens the door of the empty room, he will go on with staying in prison.

The sign on the door of the room containing the lady is true. The sign on the door of the room containing tigers is false. We don't know anything about the falsehood of the sign on the empty room The signs say respectively:

1. Room #3 is empty.
2. The tiger is in room #1.
3. This room is empty.

Exercise 6.7 *Do you know in which room the lady is now?*

Exercise 6.8 *Implement the puzzle and repeat all tasks stated in exercise 6.5.*

6.2 Finding more models: graph coloring

Typically, puzzles of this sort have a unique solution, i.e., precisely one model consistent with all conditions could be produced. We will explore now situations when many more models are stake. As an example, we should consider the problem of coloring each node of a graph with one color from a given set, such that neighbor nodes have different colors. We will employ Mace4 once again for finding models of the knowledge base [1].

Exercise 6.9 *Let us assume we are given a set of 4 nodes a, b, c, d connected by a total of 5 edges, with the edge (b, c) being the only one missing from the 4-nodes clique. We are also given a set of 3 colors $\{Red, Green, Blue\}$. Write down sentences in Propositional Logic to formalize the coloring restrictions described above, then ask Mace4 to find all models for the knowledge base. You should carefully express the following:*

- *each node has assigned at least a color*

[1] There are many more tools specialized in this task, which is fundamental in AI

- *no node can have more than one color*
- *neighbors cannot have the same color*

How many models do you get?

Exercise 6.10 *Do the same tasks as in exercise 6.9 for a graph with 10 nodes, using 6 colors. The edge set is up to you. Test the number of models produced and the running time.*

6.3 Solutions to exercises

Solution to exercise 6.1: The lady is in the first room.

Solution to exercise 6.2:

Listing 6.4: Step 1: there are some ladies in the castle

```
assign(max_seconds, 30).
set(binary_resolution).

formulas(assumptions).
 %there is a lady in room 1, 2 or 3
 l1 | l2 | l3.
end_of_list.

formulas(goals).
 %l2.
end_of_list.
```

You should get 7 models. So far, nobody tells the system there is only one lady. All we know is there exists at least one.

Solution to exercise 6.3: 3 models: lady in room #1, or in room #2, or #3.

Solution to exercise 6.4: No. We have already expresses that. Check that $p \rightarrow q$ and $\neg q \rightarrow \neg p$ are equivalent, then use the equivalence between $p \rightarrow q$ and $\neg p \vee q$.

Solution to exercise 6.5: 1 model: lady in room #1.

Solution to exercise 6.6: 2 models: lady, tiger, tiger and tiger, tiger, lady respectively.

Solution to exercise 6.7: The lady is once again in the first room.

Solution to exercise 6.8:

Listing 6.5: One lady and one tiger plus an empty room

```
assign(max_seconds, 30).

formulas(assumptions).

 %there is a lady in room 1, 2 or 3
 l1 | l2 | l3.
 %no lady in more than 1 room; the princess is unique
 l1 -> -l2.   l1 -> -l3.   l2 -> -l1.
 l2 -> -l3.   l3 -> -l1.   l3 -> -l2.

 %there is 1 tiger in one of the rooms 1, 2 or 3
 t1 | t2 | t3.

 %no tiger in more than 1 room; the tiger is unique as well
 t1 -> -t2.   t1 -> -t3.   t2 -> -t1.
 t2 -> -t3.   t3 -> -t1.   t3 -> -t2.

 %we have one empty room
 e1 | e2 | e3.
 %but no more than one
 t1 -> -t2.   t1 -> -t3.   t2 -> -t1.
 t2 -> -t3.   t3 -> -t1.   t3 -> -t2.

 %no tiger in the room where the lady stays
 l1 -> -t1.   l2 -> -t2.   l3 -> -t3.

 %the room where the lady stays is not empty
 l1 -> -e1.   l2 -> -e2.   l3 -> -e3.

 %the room where a tiger stays is not empty
 t1 -> -e1.   t2 -> -e2.   t3 -> -e3.

 %the clue on the lady's room is true; on the tiger's room is false
 %the clue on the empty room is either false or true

 %clue on door #1: room #3 is empty
 l1 -> e3.    t1 -> -e3.

 %clue on door #2: the tiger is in room #1
 l2 -> t1.    t2 -> -t1.

 %clue on door #3: this room is empty
 l3 -> e3.    t3 -> -e3.
end_of_list.

formulas(goals).
 %l1.
end_of_list.
```

Solution to exercise 6.9:

Listing 6.6: Graph coloring: 4 nodes and 5 edges

```
formulas(assumptions).
 %each node has assigned at least a color
 a_Red | a_Green | a_Blue.
 b_Red | b_Green | b_Blue.
 c_Red | c_Green | c_Blue.
 d_Red | d_Green | d_Blue.

 %no node can have more than one color
 a_Red -> -a_Blue.
 a_Red -> -a_Green.
 a_Green -> -a_Blue.

 b_Red -> -b_Blue.
 b_Red -> -b_Green.
 b_Green -> -b_Blue.

 c_Red -> -c_Blue.
 c_Red -> -c_Green.
 c_Green -> -c_Blue.

 d_Red -> -d_Blue.
 d_Red -> -d_Green.
 d_Green -> -d_Blue.

 %neighbors cannot have the same color
 a_Red -> -b_Red.  a_Red -> -c_Red.
 a_Red -> -d_Red.  a_Green -> -b_Green.
 a_Green -> -c_Green.  a_Green -> -d_Green.
 a_Blue -> -b_Blue.  a_Blue -> -c_Blue.
 a_Blue -> -d_Blue.

 b_Red -> -d_Red.  b_Blue -> -d_Blue.
 b_Green -> -d_Green.

 c_Red -> -d_Red.  c_Blue -> -d_Blue.
 c_Green -> -d_Green.
end_of_list.

formulas(goals).
end_of_list.
```

Bibliography

[1] Gheorghe Păun. *Matematica? Un spectacol!* Editura Ştinţifică şi Enciclopedică, 1988.

[2] Raymond M. Smullyan. *The lady or the tiger? : and other logic puzzles, including a mathematical novel that features Coders great discovery.* Alfred Knopf, Inc., 1982.