Department of Computer Science
Technical University of Cluj-Napoca

Introduction to Artificial Intelligence

Adrian Groza, Radu Razvan Slavescu and Anca Marginean

Contents

Lab 8

First Order Logic

Adding numbers and an object equality operator would help Prover9 express more easily some knowledge. Some more inference rules are needed in order to deal with these new elements.

> Learning objectives for this week are:
>
> 1. To get familiar with resolution and the way clauses are preprocessed before actually performing it
> 2. To see how basic arithmetic is managed in Prover9
> 3. To get familiar with paramodulation inference rule

8.1 Revisiting Resolution and Skolemization

8.1.1 Resolution

Recall the task of proving colonel West is a criminal as stated in [2], page 336.

The law says it is a crime for an American to sell weapons to hostile nations. The country Nono, an enemy of America, has some missiles. All of its missiles were sold to it be Colonel West, who is American.

We possess the following background knowledge: missiles are weapons; enemies of America are hostile.

Exercise 8.1 *Implement in Prover9 the problem about colonel West and prove he is a criminal. For every predicate, specify the meaning of each of its arguments.*

In the proof produced, we have:

```
 8 -Missile(x) | -Owns(Nono,x) | Sells(West,x,Nono).  [clausify(2)].
9 Owns(Nono,M1).  [assumption].
...
12 -Missile(M1) | Sells(West,M1,Nono).  [resolve(8,b,9,a)].
```

Clause 12 is obtained by performing a resolution over clauses 8 and 9. Here, `8,b` means the second part of clause number 8, namely `-Owns(Nono,x)` (clause parts are separated by `|`). The second sentence needed by the resolution is the first (and only) part of 9, hence the `9,a` notation. The result is clause 12.

Exercise 8.2 *Try to follow and explain at least one more resolution step in the proof.*

8.1.2 Understanding Skolemization

Now let's focus on Curiosity problem ([2], page 354). We know that:

1. Everyone who loves all animals is loved by someone.
2. Anyone who kills an animal is loved by no one.
3. Jack loves all animals.
4. Either Jack or Curiosity killed the cat, who is named Tuna.
5. Cats are animals. (background knowledge)

Question: Did Curiosity kill the cat?

Prover9 uses `all` and `exists` for the logical ∀ and ∃ respectively. By default, variables in a clause are universally quantified, so `all` could be omitted. Please be careful to the quantifier scoping: parentheses could be of great help. The best practice is probably to write `exists x ( )` and just after that to fill in the spaces inside the pparentheses

Exercise 8.3 *Implement in Prover9 the problem about who killed the cat. Try to prove the goal "Curiosity killed Tuna". Now change the goal to "There exists someone who killed Tuna" and try to see who this is. Do you get the correct answer?*

Exercise 8.4 *In the proof produced, we have:* `Animal(f1(x)) | Loves(f2(x),x)`*. What do* `f1(x)` *and* `f2(x)` *mean?*

8.2 Paramodulation and Demodulation

We consider as an example a part of a royal family. Elizabeth II is the mother of two sons: Charles and Andrew. Charles is the father of William and Harry. Elizabeth II is the Queen of UK. Prince William has become Duke of Cambridge on his wedding day.

The goal is to prove the Queen is the grandparent of the Duke of Cambridge. The implementation is given in listing 8.1.

Listing 8.1: The Queen and the Duke of Cambridge

```
set(paramodulation).

% constants: ElizabethII,
% predicate Mother(x,y): x is motehr of y

formulas(assumptions).
  Mother(ElizabethII, Charles).
  Mother(ElizabethII, Andrew).
  Father(Charles, William).
  Father(Charles, Harry).
  Father(x,y) | Mother(x,y) -> Parent(x,y).
  Parent(x,y) & Parent(y,z) -> Grandparent(x,z).
  William = DukeOfCambridge.
  Queen = ElizabethII.
end_of_list.

formulas(goals).
  Grandparent(Queen,DukeOfCambridge).
end_of_list.
```

Listing 8.2: The Queen and the Duke of Cambridge - the proof

```
1 Father(x,y) | Mother(x,y) -> Parent(x,y) # label(non_clause).  [assumption].
2 Parent(x,y) & Parent(y,z) -> Grandparent(x,z) # label(non_clause).  [assumption].
3 Grandparent(Queen,DukeOfCambridge) # label(non_clause) # label(goal).  [goal].
```

```
4 -Mother(x,y) | Parent(x,y).  [clausify(1)].
5 Mother(ElizabethII,Charles).  [assumption].
7 -Father(x,y) | Parent(x,y).  [clausify(1)].
8 Father(Charles,William).  [assumption].
10 -Parent(x,y) | -Parent(y,z) | Grandparent(x,z).  [clausify(2)].
11 William = DukeOfCambridge.  [assumption].
12 DukeOfCambridge = William.  [copy(11),flip(a)].
13 Queen = ElizabethII.  [assumption].
14 -Grandparent(Queen,DukeOfCambridge).  [deny(3)].
15 -Grandparent(ElizabethII,William).  [copy(14),rewrite([13(1),12(2)])].
16 Parent(ElizabethII,Charles).  [resolve(4,a,5,a)].
18 Parent(Charles,William).  [resolve(7,a,8,a)].
20 -Parent(Charles,William).  [ur(10,a,16,a,c,15,a)].
21 $F.  [resolve(20,a,18,a)].
```

Run the example and take a look at the proof, which is copied in Listing 8.2. Clause 15 (`15 -Grandparent(ElisabethII,William). [copy(14),rewrite([13(1),12(2)])].`) is obtained from clause 14 by applying demodulation (aka rewriting), using clause 13 and 12. As stated in the Prover9 documentation [1], demodulation is the process of using a set of oriented equations to rewrite (simplify, canonicalize) terms. The first part, `[13(1)`, tells that the left side of clause 13 (i.e., `Queen`) gets replaced by its right side (`ElizabethII`) inside clause 14. The part `12(2)` says that `DukeOfCambridge` is replaced by `William`. The numbers 1 and 2 indicate the position of rewriting (although this is not fully documented in Prover9).

The ability to deal with equality adds some power to Mace4. Another improvement would be the possibility to do simple arithmetic. By doing `set(arithmetic)`, you give Mace4 access to operators like `+,*,/` or `<.>,<=,>=`. A full description of operators is given in [1].

We consider Einstein's riddle again, but this time using arithmetic operators and properties. The implementation in `alligatorBeer2.in` follows the approach in [1]. First, we do:

```
set(arithmetic).
assign(domain_size, 5).
```

which will allow us define the "right neighbor" relation and to tell the five houses are numbered as `{0,1,2,3,4}` respectively. Then, we specify that objects in the same category are mutually distinct:

```
list(distinct).
 [Alligator,Bulldog,Cat,Donkey,Eagle]. % pets are distinct
 [Aston_Martin,Bugatti,Cadillac,Dacia,Edonis].  % cars are distinct
 ...
end_of_list.
```

To define neighbors, we write:

```
right_neighbor(x,y) <-> x+1 = y.
neighbors(x,y) <-> right_neighbor(x,y) | right_neighbor(y,x). % y left/right
```

The clues are simply written like this:

```
Austrian = Amber.
```

Exercise 8.5 *Solve the puzzle written in this new manner, by typing*
mace4 -c -f alligatorBeer1.in | interpformat
Can you tell who drinks beer?

The output contains an interpretation consisting of a set of functions and relations, which actually describe a model produced by Mace4. For example, `function(Cat,[2])` tells us "We have a cat in house 2" (which is actually the house in the middle).

If you call `interpformat` with the `tex` option (`interpformat tex`), you can generate the TeX output, which could be used further in a LaTeX file. An excerpt of the output produced by the call

`mace4 -c -f alligatorBeer1.in | interpformat tex`

is given in listing 8.3.

Listing 8.3: TeX output for Einstein's Riddle

```
\begin{table}[H]  \centering % size 5
Amber: 2 \hspace{.5cm}
Cat: 2 \hspace{.5cm}
\dots
\end{table}
```

The pdf obtained from the generated tex, which obviously needs editing, can be seen at the end of this file.

The following two examples are taken from the distribution kit of [1]. The first example asks you to show that a group where $x * x = e$ for all x is commutative. You can find the implementation in `x2.in`. Take a look at that, then generate the proof. File `x2.out` contains such a proof. Inside, you can find an example of applying an inference rule called paramodulation. This is basically the resolution modified to deal with equality.

If we look at lines 2, 3, 4 and 7:

```
2 e * x = x.  [assumption].
3 x' * x = e.  [assumption].
4 (x * y) * z = x * (y * z).  [assumption].
7 x' * (x * y) = y.  [para(3(a,1),4(a,1,1)),rewrite([2(2)]),flip(a)].
```

we can see caluse 7 is obtained by applying paramodulation over clauses 3 and 4. Inside `para(3(a,1),4(a,1,1))`, 1 refers to the left side and 2 to right side of the equality, while index `a` refers to the part of the clause. Given this, we can see that the part `x' * x` in 3 will replace the `(x * y)` part in the left side of 4. This makes `x'` to unify with `x` and `x` with `y`, getting us `(x' * x) * z = x' * (x * z)`. Further, `(x' * x)` is replaced by `e`. The next step is to apply clause 2 and simplify `e * z` to `z`; the last step is to flip the sides of eequalityand obtain `x' * (x * z) = z`, which is basically the clause 7 modulo one variable renaming. The whole bunch of steps is actually one application of paramodulation.

Exercise 8.6 *Drop the condition $x * x = e$ for all x and try to prove or disprove the group remains commutative.*

Exercise 8.7 *Can you prove $\sqrt{2}$ is not a rational number?*

Prover9, if fed the file `sqrt2.in`, can generate the proof.

Exercise 8.8 *Take a look at the input file and see how the domain is modeled. Generate the proof. Try to follow and explain at least one more paramodulation step in the proof. Delete the clause `2 != 1` from the input and see if you get the same result.*

Note: You can find many problems implemented at Thousands of Problems for Theorem Provers [3] (or TPTP for short, `http://www.cs.miami.edu/~tptp`). The program `tptp_to_ladr` might help you translate between formats.

8.3 Let's find the Wumpus!

Exercise 8.9 *Consider the Wumpus world scenario again. We assume* `(1,1)` *is safe, there is neither breeze, nor stench, nor gold there and we have a total of 3 pits, 1 wumpus and 1 gold treasure. Questions:*

1. *calculate how many different worlds are there which obey the stipulated conditions*
2. *estimate how many rules you need to write in your knowledge base to find the Wumpus if you stick with the Propositional Logic*
3. *write down a knowledge base for the same problem using now the First Order Logic and arithmetic operators and try to find the Wumpus by successively visiting safe rooms and adding the perceptions obtained there to your knowledge base.*

8.4 Solutions to exercises

Solution for exercise 8.1 is given in listing 8.4. It closely follows the solution in [2].

Listing 8.4: Colonel West is a criminal

```
% Russell, R. and Norvig, P. (2010). Artificial intelligence: A modern
% approach, 3rd Edition. pp. 336. Prentice Hall: Upper Saddle River, NJ.
%
% 1. It is a crime for an American to sell weapons to hostile nations.
% 2. The country Nono, an enemy of America, has some missiles.
% 3. All of its missiles were sold to it be Colonel West, who is American.
% 4. Missiles are weapons (background information)
% 5. Enemies of America are hostile (background information)
% Prove that West is a criminal
%
% constants: West (col. West), Nono (country), M1 (missile)
% predicate American(x): x is American
% predicate Weapon(x): x is a weapon
% predicate Sells(x,y,z): x sells object y to z
% predicate Hostile(x): x is hostile
% predicate Owns(x,y): entity x owns object y
% predicate Missile(x): x is a missile
% predicate Weapon(x): x is a weapon
% predicate Enemy(x,America): x is an enemy of America
% predicate Criminal(x): x is a criminal

set(print_gen).

formulas(assumptions).
 American(x) & Weapon(y) & Sells(x,y,z) & Hostile(z) -> Criminal(x). %1
 Owns(Nono,M1). %2
 Missile(M1). %2
 Enemy(Nono,America). %2
 Missile(x) & Owns(Nono,x) -> Sells(West,x,Nono). %3
 American(West). %3
 Missile(x) -> Weapon(x). %4
 Enemy(x,America) -> Hostile(x). %5
end_of_list.

formulas(goals).
 Criminal(West).
end_of_list.
```

Solution for exercise 8.3 is given in listing 8.5. It closely follows the solution in [2].

Listing 8.5: Curiosity killed the cat

```
% Russell, R. and Norvig, P. (2010). Artificial intelligence: A modern
% approach, 3rd Edition. pp. 354. Prentice Hall: Upper Saddle River, NJ.
%
% 1. Everyone who loves all animals is loved by someone.
% 2. Anyone who kills an animal is loved by no one.
% 3. Jack loves all animals.
% 4. Either Jack or Curiosity killed the cat, who is named Tuna.
% 5. Cats are animals. (background knowledge)
% Did Curiosity kill the cat?
%
% constants: Jack, Curiosity, Tuna
% predicate Animal(x): x is an animal
% predicate Loves(x,y): x loves y
% predicate Kills(x,y): x kills y
% predicate Cat(x): x is a cat

set(binary_resolution).

formulas(assumptions).
 all x (all y (Animal(y) -> Loves(x,y)) -> exists y (Loves(y,x))). %1
 all x (exists z (Animal(z) & Kills(x,z)) -> all y (-Loves(y,x))). %2
 all x (Animal(x) -> Loves(Jack,x)). %3
 Kills(Jack,Tuna) | Kills(Curiosity,Tuna). %4
 Cat(Tuna). %4
 all x (Cat(x) -> Animal(x)). %5
end_of_list.

formulas(goals).
 Kills(Curiosity,Tuna).
end_of_list.
```

Hint for exercise 8.4: remember Skolem functions.

Hint for exercise 8.9: $C_{13}^{3} * 13 * 15$ (positions for pits, wumpus, gold respectively).

Advocaat: 3 Amber: 2 Aston_Martin: 4 Austrian: 2 Beige: 0 Belgian: 4
Bugatti: 2 Bulldog: 4 Cadillac: 0 Cat: 2 Cider: 2 Cyan: 3 Czech: 0
Dacia: 1 Daiquiri: 4 Dane: 1 Denim: 1 Donkey: 1 Eagle: 0 Edonis: 3
Eiswein: 1 Emerald: 4 Estonian: 3 Alligator: 3 Beer: 0

neighbors:	0	1	2	3	4
0	0	1	0	0	0
1	1	0	1	0	0
2	0	1	0	1	0
3	0	0	1	0	1
4	0	0	0	1	0

right_neighbor:	0	1	2	3	4
0	0	1	0	0	0
1	0	0	1	0	0
2	0	0	0	1	0
3	0	0	0	0	1
4	0	0	0	0	0

Table 8.1: The model for Einstein's riddle produced by `interpformat tex`

Bibliography

[1] W. McCune. Prover9 and mace4. `http://www.cs.unm.edu/~mccune/prover9/`, 2005–2010.

[2] Stuart J. Russell and Peter Norvig. *Artificial Intelligence - A Modern Approach (3. internat. ed.)*. Pearson Education, 2010.

[3] G. Sutcliffe. The TPTP Problem Library and Associated Infrastructure. From CNF to TH0, TPTP v6.4.0. *Journal of Automated Reasoning*, 59(4):483–502, 2017.