Artificial Intelligence Project

Author: Nikhitha Saidugari

TABLE OF CONTENTS

Knowledge Representation and Automated Theorem Proving with PROVER9

Objective

This project focuses on utilizing Prover9, an automated reasoning tool, to conduct formal proofs based on first-order logic (FOL) representations. The approach involves:

1. Formulating logical statements: Representing the clauses in first-order logic.
2. Clause conversion: Transforming logical sentences into clause form, applying Skolemization where necessary.
3. Automated proof verification: Employing Prover9 to perform automatic refutation proofs, verifying logical conclusions.

Puzzle-Based Proof Solving

Puzzle 1: Reasoning About Dogs

1. Dogs like bones.
2. Dogs eat everything they like.

3. Max is a dog.
4. (Conclusion) Max eats bones.

Puzzle 2: Logical Deduction on Birds and Their Habitats

1. Every bird sleeps in some tree.
2. Every loon is a bird, and every loon is aquatic.
3. Every tree in which any aquatic bird sleeps is beside some lake.
4. Anything that sleeps in anything that is beside any lake eats fish.
5. (Conclusion) Every loon eats fish.

Approach:

1. Represent the clauses in First-Order Logic (FOL):
   - Identify predicates and constants.
   - Represent relationships and implications using logical symbols.
2. Convert the logic sentences to Clause Form or Conjunctive Normal Form (CNF):
   - Eliminate implications.
   - Standardize variables.
   - Skolemize (eliminate existential quantifiers).
   - Convert to conjunctive normal form (CNF).
3. Use Prover9 for refutation:
   - Input the clause forms as assumptions and negated goal.
   - Analyze the Prover9 output for the proof.
4. Write the report:
   - Include predicate form, clause form, assumptions, goal, and proof.

Puzzle 1

Problem Statement

- Dogs like bones.
- Dogs eat everything they like.
- Max is a dog.
- (Conclusion) Max eats bones.

FOL Representation:

Predicates:

- Dog(x) : x is a dog.
- Likes(x, y) : x likes y.
- Eats(x, y) : x eats y.
- Bone(y) : y is a bone.
- x = Max : Max is an individual constant.

1. First Order Logic Statements:

   1. ∀x (Dog(x) → Likes(x, Bone))
   2. ∀x ∀y (Dog(x) ∧ Likes(x, y) → Eats(x, y))
   3. Dog(Max)
   4. Goal: Eats(Max, Bone)

2. Clause Form or Conjunctive Normal Form:

   1. ¬Dog(x) ∨ Likes(x, Bone)
   2. ¬Dog(x) ∨ ¬Likes(x, y) ∨ Eats(x, y)
   3. Dog(Max)
   4. Goal: ¬Eats(Max, Bone)

3. Assumptions and Goal

Assumptions: The above clauses (1-3).

Goal: To derive a contradiction from the negation of the conclusion (¬Eats(Max, Bone)).

4. Prover9 Proof

Input File for Prover9:

-Dog(x) | Likes(x, bone).

-Dog(x) | -Likes(x, y) | Eats(x, y).

Dog(max).

Goal:

Eats(max, bone).

Expected Proof:

Prover9 will infer:

1. From clause 3, Max is a dog.
2. Using clause 1, since Max is a dog, Max likes bones.
3. Using clause 2, since Max is a dog and likes bones, Max eats bones.
   The conclusion "Max eats bones" will be derived, leading to a contradiction when ¬Eats(Max, Bone) is assumed.

Puzzle 1 Actual Prover9 Proof

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 80268 was started by saikr on Sai_ak,
Tue Dec 10 21:41:41 2024
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.01 (+ 0.00) seconds.
% Length of proof is 9.
% Level of proof is 3.
% Maximum clause weight is 0.
% Given clauses 0.

1 Eats(max,bone) # label(non_clause) # label(goal).  [goal].
2 Dog(max).  [assumption].
3 -Dog(x) | Likes(x,bone).  [assumption].
4 -Dog(x) | -Likes(x,y) | Eats(x,y).  [assumption].
5 -Likes(max,x) | Eats(max,x).  [resolve(2,a,4,a)].
6 -Eats(max,bone).  [deny(1)].
7 -Likes(max,bone).  [resolve(5,b,6,a)].
8 Likes(max,bone).  [resolve(2,a,3,a)].
9 $F.  [resolve(7,a,8,a)].

============================== end of proof ==========================
```

Puzzle 2

Problem Statement

- Every bird sleeps in some tree.
- Every loon is a bird, and every loon is aquatic.
- Every tree in which any aquatic bird sleeps is beside some lake.
- Anything that sleeps in anything that is beside any lake eats fish.
- (Conclusion) Every loon eats fish.

FOL Representation

Predicates:

- Bird(x) : x is a bird.
- Loon(x) : x is a loon.
- Aquatic(x) : x is aquatic.
- Tree(y) : y is a tree.
- SleepsIn(x, y) : x sleeps in y.
- Beside(y, z) : y is beside z.
- Lake(z) : z is a lake.
- Eats(x, y) : x eats y.
- Fish(y) : y is fish.

1. First Order Logic Statements:

1. $\forall x\ (Bird(x) \rightarrow \exists y\ (Tree(y) \land SleepsIn(x, y)))$
2. $\forall x\ (Loon(x) \rightarrow Bird(x) \land Aquatic(x))$
3. $\forall x\ \forall y\ (Aquatic(x) \land SleepsIn(x, y) \rightarrow \exists z\ (Lake(z) \land Beside(y, z)))$
4. $\forall x\ \forall y\ \forall z\ (SleepsIn(x, y) \land Beside(y, z) \land Lake(z) \rightarrow Eats(x, Fish))$
5. Goal: $\forall x\ (Loon(x) \rightarrow Eats(x, Fish))$

2. Clause Form or Conjunctive Normal Form:

1. $\neg Bird(x) \lor Tree(f(x))$
2. $\neg Bird(x) \lor SleepsIn(x, f(x))$
3. $\neg Loon(x) \lor Bird(x)$
4. $\neg Loon(x) \lor Aquatic(x)$
5. $\neg Aquatic(x) \lor \neg SleepsIn(x, y) \lor Lake(f(x, y))$
6. $\neg Aquatic(x) \lor \neg SleepsIn(x, y) \lor Beside(y, f(x, y))$
7. $\neg SleepsIn(x, y) \lor \neg Beside(y, z) \lor \neg Lake(z) \lor Eats(x, Fish)$
8. $\neg Loon(x) \lor Eats(x, Fish)$

3. Assumptions and Goal

Assumptions: The above clauses (1-7).

Goal: To derive clause 8 ($\neg Loon(x) \lor Eats(x, Fish)$).

4. Prover9 Proof

Input File for Prover9:

```
 -Bird(x) | Tree(f1(x)).
-Bird(x) | SleepsIn(x, f1(x)).
-Loon(x) | Bird(x).
-Loon(x) | Aquatic(x).
-Aquatic(x) | -SleepsIn(x, y) | Lake(f2(x, y)).
-Aquatic(x) | -SleepsIn(x, y) | Beside(y, f2(x, y)).
```

-SleepsIn(x, y) | -Beside(y, z) | -Lake(z) | Eats(x, fish).
Goal:
-Loon(x) | Eats(x, fish).
Expected Proof:
Prover9 will:

1. Use clause 3 to infer that loons are birds.
2. Use clauses 1 and 2 to show that every bird (and thus every loon) sleeps in some tree.
3. Use clauses 5 and 6 to show that every tree in which an aquatic bird sleeps is beside some lake.
4. Use clause 7 to conclude that anything sleeping beside a lake eats fish.
5. Derive the conclusion that every loon eats fish.

Puzzle 2 Actual Prover9 Proof

```
============================== prooftrans ============================
Prover9 (32) version Dec-2007, Dec 2007.
Process 82568 was started by saikr on Sai_ak,
Tue Dec 10 21:44:22 2024
The command was "/cygdrive/c/Program Files (x86)/Prover9-Mace4/bin-win32/prover9".
============================== end of head ===========================

============================== end of input ==========================

============================== PROOF =================================

% -------- Comments from original proof --------
% Proof 1 at 0.00 (+ 0.01) seconds.
% Length of proof is 20.
% Level of proof is 6.
% Maximum clause weight is 0.
% Given clauses 0.

1 -Loon(x) | Eats(x,fish) # label(non_clause) # label(goal).  [goal].
2 -Loon(x) | Bird(x).  [assumption].
4 -Bird(x) | SleepsIn(x,f1(x)).  [assumption].
5 Loon(c1).  [deny(1)].
6 -Loon(x) | Aquatic(x).  [assumption].
8 -Loon(x) | SleepsIn(x,f1(x)).  [resolve(2,b,4,a)].
9 Aquatic(c1).  [resolve(5,a,6,a)].
10 -Aquatic(x) | -SleepsIn(x,y) | Lake(f2(x,y)).  [assumption].
11 -Aquatic(x) | -SleepsIn(x,y) | Beside(y,f2(x,y)).  [assumption].
12 SleepsIn(c1,f1(c1)).  [resolve(8,a,5,a)].
13 -SleepsIn(x,y) | -Beside(y,z) | -Lake(z) | Eats(x,fish).  [assumption].
14 -SleepsIn(c1,x) | Lake(f2(c1,x)).  [resolve(9,a,10,a)].
15 -SleepsIn(c1,x) | Beside(x,f2(c1,x)).  [resolve(9,a,11,a)].
16 -Beside(f1(c1),x) | -Lake(x) | Eats(c1,fish).  [resolve(12,a,13,a)].
17 -Eats(c1,fish).  [deny(1)].
18 -Beside(f1(c1),x) | -Lake(x).  [resolve(16,c,17,a)].
19 Lake(f2(c1,f1(c1))).  [resolve(14,a,12,a)].
20 -Beside(f1(c1),f2(c1,f1(c1))).  [resolve(18,b,19,a)].
21 Beside(f1(c1),f2(c1,f1(c1))).  [resolve(15,a,12,a)].
22 $F.  [resolve(20,a,21,a)].

============================== end of proof ==========================
```

Summary

Both puzzles can be solved with Prover9 by following these steps:

- Express the problem using predicate logic.
- Transform the predicates into their equivalent clause form.
- Use the assumptions and the negated conclusion to run the proof in Prover9. Deriving a contradiction validates the conclusion.