Rajul Ramchandani & Conan Zhang

Professor Thomas Henderson

CS 4300 - 001

6 October 2016

Assignment A4a: Resolution Theorem Proving

1. INTRODUCTION: RAJUL

Resolution Theorem Solving is iteratively applying a knowledge base to solve and simplify given theorem in Conjunctive Normal Forms (CNFs). A CNF is basically literals that are simply an AND of ORs. As a normal form, it is useful in Resolution Theorem Proving as well as later build a smarter agent for the Wumpus World.

The Resolution Theorem Solver takes in a knowledge base and a theorem then returns whether or not the theorem can be proved by the given knowledge base. The clause produced by a resolution rule is called a resolvent. This Resolution Theorem Solver can later be given all possible frontier possibilities along with an existing knowledge base for an agent and essentially use these logical statements to help the agent make informed decisions in the Wumpus World. The implementation of this solver brings about a few questions:

1. What is the relation between number of clauses in the KB, to the number of sentences produced ?
2. What is the relation between number of clauses in the KB and the loop iterations needed?

2. METHOD: CONAN

Our Resolution Theorem Prover uses proof by contradiction to prove a certain theorem with a given knowledge base (KB). Thus, the first thing it does is insert the negation of the

theorem into the KB. Afterwards, it compares every clause to every other clause in the KB to determine the resolvents. If our resolvents contains an empty clause, then the theorem is proven. We keep looping through every clause in the KB, if the resolvents from that iteration is a subset of our KB, than the theorem cannot be proven.

To answer the questions in our Introduction, we use rewrite rules as test cases to gather data. For each case, we calculate the number of sentences produced and the relation to number of clauses in our initial KB. We also time each case to find relations between the number of disjuncts and initial clauses to time taken. The rewrite rules used to test are the commutativity of OR, associativity of OR, contraposition, and De Morgan's Law.

3. VERIFICATION OF PROGRAM: RAJUL

Case 1: **Modus Ponens**

1) (-1, 2) 2) (1) Thm: 2

Hand Solved Result:

1) (-1, 2) 2) (1) 3)(-2) (Negate theorem and add to sentences)

Running the algorithm by resolving a pair of the rules above and then union it with set New.

Pair	Resolvents	New
(1,2)	{{2}}	{{2}}
(1,3)	{{-1}}	{{2}{-1}}
(2,3)	{}	{{2}{-1}}

Table 1

Running the algorithm by resolving a pair of the rules above and then union new with sentences.

Adding(union) New to the given sentences to form new set :

1) (-1, 2) 2) (1) 3) (-2) 4) (2) 5) (-1)

Pair	Resolvents	New
(1,2)	{{2}}	{{2}}
(1,3)	{{-1}}	{{2}{-1}}

(1,4), (1,5), (2,3), (2,4)	{}	{{2}{-1}}
(2,5)	{{}}	DONE!

Table 2

Here, we end after finding the empty set inside of Resolvents. This denotes that some pair was fully resolved with no other terms left. Therefore, this theorem is proved.

Solved by Code:

When running the same problem through the algorithm written, we get

[Pair]	**[Resolvents]**	**[New]**
[1] [2]	[2]	[2]
[1] [3]	[-1]	[2 , -1]
[2] [3]	[]	[2, -1]
[1] [2]	[2]	[2]
[1] [3]	[-1]	[2, -1]
[1] [4]	[]	[2, -1]
[1] [5]	[]	[2, -1]
[2] [3]	[]	[2, -1]
[2] [4]	[]	[2, -1]
[2] [5]	[]	[DONE]

Here, the program ends after finding empty set inside of new. So the output above from the program is exactly the same as the handwritten version. This proves the correctness of the program. (code for above output found in RTP function)

Case 2: **Empty KB**

No KB and Thm : (1, -1) ->

Hand Solved Result:

1) 1 2) -1 (Negate theorem and add to sentences)

Running the algorithm by resolving a pair of the rules above and then union it with set New.

Pair	**Resolvents**	**New**
(1,2)	{{}}	DONE

Solved by Code:

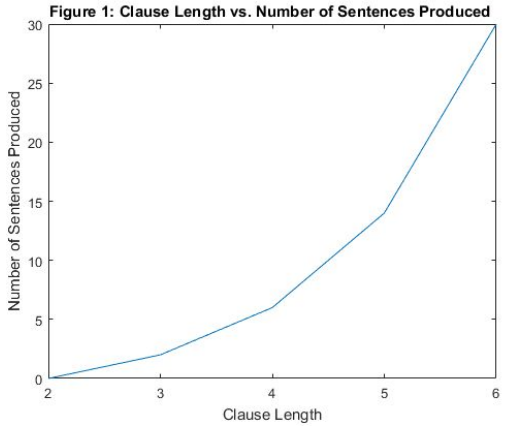
When running the same problem through the algorithm written, we get

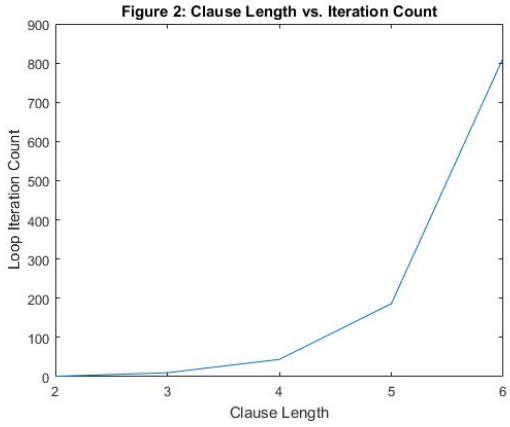
[Pair]	**[Resolvents]**	**[New]**
[1] [2]	[ ]	[DONE]

Here, the program ends after finding empty set inside of new. So the output above from the program is exactly the same as the handwritten version. This proves the correctness of the program. (code for above output found in RTP function)

4. DATA AND ANALYSIS: CONAN

For the figures below, a clause with length two to six was taken and run against the algorithm. By fixing one variable, the trends of the other variable could be observed easily. From Figure 1 and 2 below we see the trends for sentences produced and iteration count while holding clause length at a fixed value. The shape of the curves also seem to suggest the complexity of this algorithm. The complexity is quadratic.





5. INTERPRETATION: RAJUL

From the above data and experimentation, the number of sentences produced increases with the number of clauses in the KB. Here, the number of iterations also understandably increased.

It was also observed that if the number of clauses in KB was limited to 1, but the length of that clause was increased, the number of iterations as well as number of clauses produced increases significantly.

It is clear that with an increased number of a clauses, the number of iterations as well as the the clauses produced were not affected as much.

This is fairly inaccurate as the numbers that were contained in these clauses were vital to this study as well. If the correct combination of numbers was in these clauses, the time taken to finish the algorithm could be significantly less as compared to an unsuccessful case.

6. CRITIQUE: CONAN

Our algorithm used several slow functions, such as an O(n^2) remove duplicates function. These could be improved upon to increase the speed of the Resolution Theorem Prover. More data could help us draw stronger relations between number of clauses, sentences produced, and iteration count, but with the sample set we tested against it is clear that the number of clauses has a strong influence on the other variables.

7. LOG: RAJUL & CONAN

Keep a log of the time spent on each problem and include it in the report.

Rajul: I spent 6 hours on A4a and 6 hours on the Lab Report.

Conan: I spent 4 hours on A4a and 6 hours on the Lab Report.