A3 – CS4300 Assignment A3 Lab Report

Resolution Theorem Proving

By Haochen Zhang & Tim Wei
9/27/2017

1. Introduction

Two functions are written for this assignment, *BR_gen_KB* and *CS4300_RTP*. *BR_gen_KB* is a function that will populate a knowledge base with the rules of the Wampus World. *CS4300_RTP* is an implemented version of the resolution algorithm from the book. We know that the resolution algorithm is a costly algorithm, so with *BR_gen_KB* and *CS4300_RTP*. *BR_gen_KB*, we want to answer this question:

- Can *CS4300_RTP* handle a knowledge base as big as the knowledge base *BR_gen_KB* generates in a short amount of time?

2. Method

For *BR_gen_KB*, the logic that generates the knowledge base is based on 6 main rules.

a. Pit and Breeze relations: 64 (corners: 12, edges: 32, centers: 20)
b. Wampus and Stench relations: 64 (same as above)
c. Must be one Wampus: 15 + 14 + … + 1 = 120 + 1 = 121(16 cells * 15 cells should not have W)
d. Must be one Gold: 121(the same as above)
e. If there is a pit, no Wampus can be in that cell: 16
f. If there is a pit, no Gold can be in that cell: 16 (same as above)

Rule a

For a cell in the center (e.g. $Cell_{22}$), there is 4 connected cells around it. The sentence would be:

$$B_{22} \Leftrightarrow P_{12} \vee P_{21} \text{ v } P_{13} \vee P_{31}$$

This will generate 5 sentences in CNF which are

$$-B_{22} \vee P_{12} \vee P_{21} \vee P_{13} \vee P_{31}$$
$$-P_{12} \vee B_{22}$$
$$-P_{12} \vee B_{22}$$
$$-P_{12} \vee B_{22}$$
$$-P_{12} \vee B_{22}$$

For the cells in the corner, there will be only 3 sentences in CNF since there is only 2 cells that are adjacent to the cell we are looking at. For the cells on the edges, there will be 4 sentences in CNF. Thus, there will be 5 * 4 + 4 * 8 + 3 * 4 = 64 sentences in CNF.

Rule b

The same as ***Rule a*** except it is a different label.

Rule c

For a cell that has a Wampus label on it, the other 15 cells should not have that label. We have

$$W_{11} \Rightarrow -W_{21} \vee -W_{31} \vee -W_{41} \vee \ldots \vee -W_{44}$$

It will be separated into 15 sentences in CNF:

$$-W_{11} \vee -W_{21}$$
$$-W_{11} \vee -W_{31}$$
$$\vdots$$
$$-W_{11} \vee -W_{44}$$

For the next cell, for example W_{21}, there will be duplicate sentences since $-W_{11} + -W_{21}$ is equal to $-W_{21} + -W_{11}$, there will be only 14 sentences in CNF. We get 15 + 14 + ... + 1 = 120 sentences in CNF after we go through all the cells. Then, we need to add one more sentence that are conjunction of all the cells since there must be one Wampus in the board:

$$W_{11} \vee W_{21} \vee ... \vee W_{34} \vee W_{44}$$

Thus, we get 121 sentences in CNF.

Rule d

The same as ***Rule c*** except it is a different label.

Rule e

For a cell with a Pit label on it, it cannot be a Wampus label on it. So,

$$P_{11} \Leftrightarrow -W_{11}$$

It can be transform into

$$-P_{11} \vee -W_{11}$$

Since there is 16 cells and each cell will have one sentence, there are totally 16 sentences in CNF.

Rule f

The same as ***Rule e*** except it is a different label.

For *CS4300_RTP*, we used the algorithm from the book:

function PL-RESOLUTION(*KB*,*α*) returns true or false
  inputs: *KB*, the knowledge base, a sentence in propositional logic
    α, the query, a sentence in propositional logic
clauses ← the set of clauses in the CNF representation of *KB* ∧ ¬*α*
new ← {}
loop do
  for each pair of clauses *Ci*, *Cj* **in** clauses **do**
    resolvents ← PL-RESOLVE(*Ci*, *Cj*)
    if *resolvents* contains the empty clause then return *true*
    new ← *new* ∪ *resolvents*
  if *new* ⊆ clauses **then return** *false*
  clauses ← *clauses* ∪ *new*

In our implementation of *CS4300_RTP*, we use a empty clause ([]) to represent true, and *clauses* to represent false. The reason that *clauses* is chosen is because that in the case that the function return false, we can trace the computation to see what went wrong.

Some optimization are added to improve performance:

- Clauses generated from the theorem are put in the front of the clauses list.
- Skips the *Cj* that have been paired in the last while loop iteration.
- Skips the *new* that have been check for duplication in the last while loop iteration.

To answer the question posted in the introduction, we will perform three tests:

- Using $P_{11} \cup$ KB, prove the theorem B_{21}.
- Using $P_{11} \cup$ KB, prove the theorem B_{31}.
- Using $B_{11} \cup$ KB, prove the theorem $P_{21} \vee P_{12}$.

If the function run for more than 5 minutes, we deem it as not capable of computing such theorem with *BR_gen_KB*.

3. Verification of Program

In our *BR_gen_KB*, we have 402 sentences which is the right number that should be generated. In our code, we generate one rule at a time because it is easy to debug. After we run *BR_gen_KB()*, we have checked all the edge cases such as sentences #1, #64, # 65, # 128, #129, # 249, #250, #370, #371, #386, #387 and #402. All these clauses matches our manual results.
The value of these cells in KB are listed in the following:

#1:	$-B_{11}$ P_{21} P_{12}
#64:	$-P_{43}$ B_{44}
# 65:	$-S_{11}$ W_{21} W_{12}
# 128:	$-W_{43}$ S_{44}
#129:	W_{11} W_{21} W_{31} W_{41} W_{12} W_{22} … W_{34} W_{44}
#130:	$-W_{11}$ $-W_{21}$
# 249:	$-W_{34}$ $-W_{44}$
#250:	G_{11} G_{21} G_{31} G_{41} G_{12} G_{22} … G_{34} G_{44}
#251:	$-G_{11}$ $-G_{21}$
#370:	$-G_{34}$ $-G_{44}$
#371:	$-P_{11}$ $-W_{11}$
#372:	$-P_{11}$ $-G_{11}$
#401:	$-P_{44}$ $-W_{44}$
#402:	$-P_{44}$ $-G_{44}$

CS4300_RTP was given a small KB:

```
>> KB(1).clauses = [-1,2,3,4];
 KB(2).clauses = [-2];
 KB(3).clauses = [-3];
 KB(4).clauses = [1];
```

Test with theorem [4] yield:

```
>> Sr = CS4300_RTP(KB,thm,vars)
Sr = []
```

Which match the hand derived result:

R_1 = [-1,2,3,4]
R_2 = [-2]
R_3 = [-3]
R_4 = [1]

R_5 = [-4]
R_6 = [-1,3,4] (R_1 + R_2)
R_7 = [-1,4] (R_3 + R_6)
R_8 = [-1] (R_4 + R_7)
R_9 = [] (R_5 + R_8)

Test with theorem [3] yield:

```
>> Sr = CS4300_RTP(KB,thm,vars)
Sr = [-3],
     [-1, 2, 3, 4],
     [-2],
     [-3],
     [1],
     [-1, 3, 4],
     [2, 3, 4],
     [-1, 4],
     [2, 4],
     [3, 4],
     [4]
```

Which is correct because the theorem contradicts KB.

4. Data and Analysis

In the three tests:

- Using $P_{11} \cup$ KB, prove the theorem B_{21}.
- Using $P_{11} \cup$ KB, prove the theorem B_{31}.
- Using $B_{11} \cup$ KB, prove the theorem $P_{21} \vee P_{12}$.

Only the first test returned in 5 minutes. The other could not finish in time. Using the debug tool in Matlab, we found that both of them generates more than 10,000 new clauses during the second iteration of the while loop.

5. Interpretation

CS4300_RTP is very slow with a KB as large as *BR_gen_KB*. It can only prove simple theorem that generates empty clauses in the first iteration of the loop (or early in the second iteration). If the theorem is false or have multiple literals, it will loop into the second iteration which generates unsustainable amount of new clauses. In future agents, we should combine search and resolution to achieve better performance.

6. Critique

The speed of *CS4300_RTP* is not fast enough to use on Wampus Word agent. It is powerful only on a knowledge base that is smaller enough. For the knowledge base that are strongly connected with one another, *CS4300_RTP* will become significant slow once it reach the second circle since optimization are not applicable anymore. Since *CS4300_RTP* cost so much time, we are not able to test more theorem.

7. Log

Haochen Zhang(Section 2, 4, 6)
A total of 8 hours was spent performing the experiment in Matlab.
A total of 3 hours was spent performing the experiment in writing the report.

Tim Wei (Section 1, 3, 5)
A total of 10 hours was spent performing the experiment in Matlab.
A total of 3 hours was spent performing the experiment in writing the report.