Monish Gupta (Evens) and Eric Waugh (Odds)

U1008121 u0947296

CS4300

September 22 2017

Assignment 3 : Lab Report

1. *Introduction*

   Two functions where one generates a general wumpus world rule set as CNF structure and a function that takes in a knowledge base and a theorem and tries to show that the KB entails the theorem and does this using resolution theorem proving. Its evident that RTP uses a brute-force style method when addressing sentences within its knowledge base. As a result, the Resolution Theorem Prover becomes slower to come to a conclusion. It becomes necessary to find out a reasonable limit to the knowledge base size to which RTP can still be useful. What is this limit?

2. *Method*

   <u>CS4300_BR_gen_KB</u> - This function is setup to generate rules about Wumpus World. A total of 402 sentences were generated to build a comprehensive way to describe the behavior of Wumpus World. These rules include cell coordinates and a set of 5 special labels : B, G, P, S, W standing for “Breeze” , “Gold” , “Pit”, “Stench” , and “Wumpus” accordingly. In generating the rules, the neighbors of each individual cell must be considered. In this case, a connectivity graph helps to enumerate neighbors of cells in order to add rules relevant to each cell. In the generation of the connectivity graph, one can observe that corners on the 4x4 grid have 2 connections, edges cells have 3 connections and interior cells have 4 connections. Since there are 4 corner cells, 8 edge cells, and 4 interior cells there are 48 connections to consider (2*4 + 8*3 + 4*4). The rules generated are as follow :

   > *If there is a breeze in a cell then there must be at least one pit in its neighboring cells.* **64 sentences**

If there is a stench in a cell then there must be a wumpus in at least of of that cell's neighboring cells. **64 sentences**

There must be one wumpus on the board. **120 sentences**

There must be one gold on the board. **120 sentences**

The wumpus must be on a valid space (The wumpus must not be positioned on a cell which contains a pit). **16 sentences**

The gold must be on a valid space (The gold must not be positioned on a cell which contains a pit). **16 sentences**

There must be gold on the board at some cell. **1 sentences**

There must be wumpus on the board at some cell. **1 sentences**

CS4300_RTP - This function is a general purpose Resolution Theorem Prover that attempts to resolve a CNF theorem with an accompanying knowledge base and return whether the theorem is provable or not (represented with an empty or complete set). In order to see if a theorem is provable its clauses are first insert, negated, into the knowledge base that is passed in. This is because the Theorem Prover algorithm attempts to prove a theorem through a proof by contradiction. After the negated clauses are inserted, a 'new' list is created to track new sentences added to the knowledge base. Next, clauses are looped through in pairs in a brute force method. Each pair is processed within a function that finds tautologies and collapses them into possible new clauses; these clauses are inserted in 'new'. If new is a subset of our knowledge base then new clauses haven't been come across and the theorem can't be proved. Otherwise new clauses are added to the knowledge base and will be processed again. If in the chance that, when resolving two clauses, all of the literals within the two clauses have been collapsed then the theorem has been proved and can be returned in the form of an empty set.

3. *Verification of Program*

To verify that we have the correct rule set for CS4300_BR_gen_KB we add up all the rules that are covered in the method section which would be 64 + 64 + 120 + 120 + 16 + 16 + 2 = 402 rules. There are 64 rules for the breeze logic having a neighboring pit, 64 similar rules but for stenches to wumpus instead, 120 rules that there is wumpus only on one spot on the board, a similar 120 rules that there is only gold on one spot of the board, 16 rules that there is no wumpus on a pit, another similar 16 for no gold on a pit. Then there are are extra two sets of 16 rules that give more information that if a cell has a pit then all of its neighbors must have breezes and a similar rule set for wumpus' and stenches.

Workspace

Name	Value
KB	1x402 struct
KBi	1x402 struct
vars	1x80 struct

Size of returned elements from generate_kb

Variables - KBi

KBi

1x402 struct with 1 field

Fields	clauses
1	[-34,1]
2	[-37,1]
3	[-1,34,37]
4	[-33,2]
5	[-35,2]
6	[-38,2]
7	[-2,33,35,...
8	[-34,3]
9	[-36,3]
10	[-39,3]
11	[-3,34,36,...
12	[-35,4]
13	[-40,4]
14	[-4,35,40]
15	[-33,5]
16	[-38,5]
17	[-41,5]
18	[-5,33,38,...
19	[-34,6]
20	[-37,6]
21	[-39,6]

First 21 rules in KBi

To verify CS4300_RTP we ran it with the example provided where DP would be the knowledge base and we are trying to prove and thm is the theorem we are trying to prove. When we run this we get the expected result of empty since 4 can be proved through contradiction. The opposite theorem of -4 for thm also gives the correct output of it not being empty since you can not prove it by contradiction.

Input:

DP(1).clauses = [-1,2,3,4];

DP(2).clauses = [-2];

DP(3).clauses = [-3];

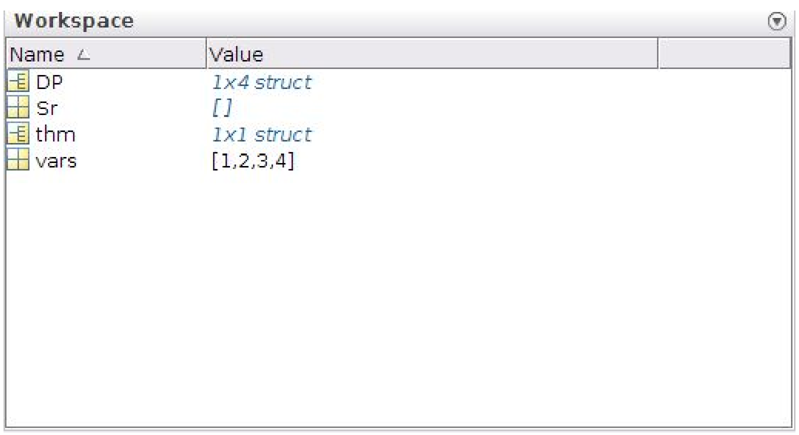
DP(4).clauses = [1];

thm(1).clauses = [4];

vars = [1,2,3,4];

Call: Sip = CS4300_RTP(DP,thm,vars);

Sip = []



Workspace after running said code

4. *Data and Analysis*

```
K>> clear all;
n = 5;
vars = 1:n+2
 for i = 1 : n
DP(i).clauses = [-i,i+1];
end
thm(1).clauses = [2,6,1];
CS4300_RTP(DP,thm,vars);
```

Sample Driver Code

Sample Driver Code for Clause Length 2 (-i, i+1)

n	3	4	5	6	7	8	9
New sentences	4	7	15	20	26	33	41

Sample Driver Code for Clause Length 3 (-i, i+1, -(i+2))

n	3	4	5	6	7	8	9
New Sentences	9	21	48	94	171	302	521

5. *Interpretation*

When looking at this data you can see that when the length of clauses was equal to 2 it grew at a more linear rate, when the length of clause is 3 it grew at a more exponential rate. We can interpret this as the length of clauses getting larger it will create more new combinations of sentences that shorter length clauses will. It will take more time to break these combinations down or find a point where nothing new is being added. This obviously concludes that there is some realistic limit to how long and how many clauses we can have where this will run at an optimal speed to find a solution. When considering this for wumpus world in its current state it does not seem like it would be fast enough to give the agent the data he needs right away so he would have to wait a sizable amount of time before each move.

6. *Critique*

A few improvements can be made within the methodology of this experiment. First, it seemed that ‘Vars’ was implementation specific. There was a simple method to transform cell numbers into their corresponding coordinates on the cell board. Not only is this operation constant time, it doesn't necessitate an extra list or accompanying searches to get the correct data. The representation of CNF as a vector of structs made the need to write code that needn't be written if the data represented in another way such as a set length matrix. These functions are fine tuned and specially optimized for the data structure MATLAB specializes in. With a home-made structure these optimizations are given up and code to perform set operations become recondite.

7. *Log*

   Eric : CS4300_BR_gen_KB - 9 hours
   CS4300_RTP - 6 hours
   CS4300_PL_Resolve - 1 hour
   Lab Report - 5 hours

   Monish : CS4300_BR_gen_KB - 9 hours
   CS4300_RTP - 6 hours
   CS4300_PL_Resolve - 1 hour
   Lab Report - 5 hours