EVALUATION FOR CLINGO ASP SOLVER

Done By

Muhammad Adeel Zafar (216047144)

Amirhossein Razavi (216715963)

Arya Mazloomi (217418609)

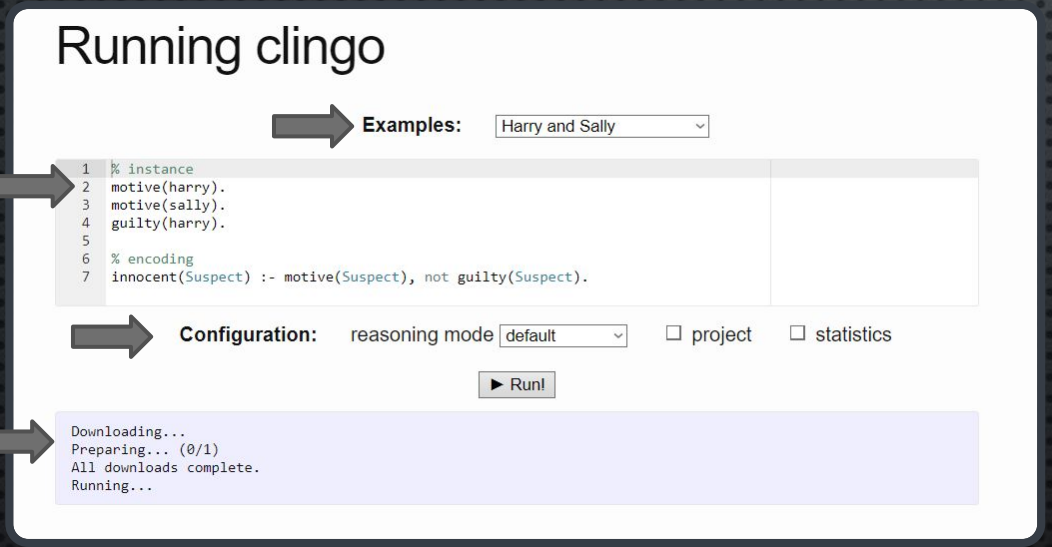
Akaf Khurshid (217181041)

WHAT IS CLINGO?

AS PRESENTED BY: AMIRHOSSEIN RAZAVI

- Clingo is an ASP Solver which lets its users solve logic-based problems.
- ASP here stands for Answer Set Programming and acts as a modeling language of sorts to help users deal with combinatorial problems.
- Usually, these problems entail dealing with the grouping, ordering and assignment of different object sets under specific conditions.

INTRODUCTION TO CLINGO AND ITS FEATURES



- CLINGO IS AN ASP SOLVER DEVELOPED AND DISTRIBUTED BY POTASSCO, THE POTSDAM ANSWER SET SOLVING COLLECTION, BUNDLES TOOLS FOR ANSWER SET PROGRAMMING DEVELOPED AT THE UNIVERSITY OF POTSDAM.
- THE BROWSER VERSION OF CLINGO IS DIVIDED INTO FOUR SECTIONS. THE TOPMOST SECTION IS A SMALL REPOSITORY OF VARIOUS EXAMPLE PROBLEMS.

- The second section of clingo involves the coding block. This is where problems are written out using clingo's programming language.
- The exact details of the syntax will be discussed later on in the presentation.
- The section after, includes a reasoning mode selector, which lets users choose from one of four possible solving modes (default, brave, cautious and enumerate all).
- The third section also contains optional checkboxes used to hide or display additional data relevant to the problem being solved, if this data is required.

- The last section of clingo is the result section.
- This section goes over the various solutions found by clingo for any given problem.
- In addition to this, as can be seen in the image to the right, clingo is capable of offering optional statistics if the user deems such data is relevant to be studied.

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.005s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 0.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0        (Eliminated:    0 Frozen:    0)
Constraints  : 0        (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

Evaluating Clingo and our methodology

- To have a better understanding of clingo's usability, the team has studied its syntaxes and ease of use
- In order to test the capabilities of clingo, the team has chosen a selection of problems to be used to assess the efficiency of the software when it comes to handling different logical problems.
- The problems chosen were altered in some cases to increase their complexity and test for potential changes in results given the increased number of details involved.
- These same problems were then also used as a basis of comparison for another ASP solver (smodels), and the results of both software were compared to check for differences.

CLINGO SYNTAX AND EASE OF USE
AS PRESENTED BY: AKAF KHURSHID

- To translate a logic problem into Clingo starts with knowing the syntax
- After every statement in Clingo finishes with '.'
- Clingo does not care about white space between statements
- The order of where the statement are declared does not matter in Clingo

	True	False	If	And	Or	Iff	Negation	Strong Negation
Clingo			:-	,	;		not	-
Logic	$\top$	$\bot$	$\rightarrow$	$\wedge$	$\vee$	$\leftrightarrow$	$\neg$	~

FACT & RULES

- To declare a fact in Clingo you simply write the fact in Clingo
- Anything Declared as a fact is said to be held true for the entire problem

```
1  p.
2  human(bob).
```

- Facts are the knowledge base for Clingo
- To declare a rule such as 'if bob is a human, then bob can't fly' in Clingo

```
1  human(bob).
2  - fly(X) :- human(X).
```

- The 'X' in '-fly(X)' and 'human(X)' states any member of '-fly' and 'human

CHOICE RULE

- JUST USING FACTS AND RULES CAN BE LIMITING AS IT THERE ARE NOT MULTIPLE ANSWERS
- CHOICE RULE ALLOWS US TO HAVE DISJUNCTION WHICH CAN LEAD TO MULTIPLE ANSWERS
- TO DECLARE A CHOICE RULE IN CHLINGO, SIMPLY WRITE THE FACT/RULE WITH CURLY BRACKETS

```
1  {human(bob)}.
2  {-fly(X)} :- human(X).
```

```
1  {human(bob)}.
2  {-fly(X)} :- human(X).
```

Answer 1

Answer 2

human(bob)

Answer 3

human(bob)
-fly(bob)

CONSTRAINTS

- INTEGRITY CONSTRAINTS ALLOWS US TO PUT CONSTRAINTS ON THE ANSWER SETS PRODUCED BY CLINGO
- SIMPLY DONE BY WRITING ':-' BEFORE THE CONSTRAINTS FOR THE ANSWER SETS IN CLINGO
- CONSTRAINTS ARE USEFUL AS NOW WE CAN DECLARE THAT 'EITHER A OR B MAY BE TRUE HOWEVER BOTH CAN'T BE TRUE'

```
1  {a}.
2  {b}.
3  :- a, b.
```

- STATEMENT 3 DECLARES THAT THE ANSWER SET CAN'T HAVE A AND B

RESULTS OF STUDY AND WHAT THEY MEAN
AS PRESENTED BY: MUHAMMAD ADEEL ZAFAR

- As mentioned before, our study was based upon tests run using a set of logical problems.
- Going over these problems, we found a specific trend in terms of what clingo is capable of and where its limitations lay.
- In this section, we'll be looking over these results and what they mean. As well as discussing reasons why certain problems ended in certain results.

PROBLEM 1 (DEFAULT)

- Our first problem is relatively simple. It involves a case of guilty and innocent from three possible culprits.
- In this case, the image on the right shows the default functioning mode of clingo, which results in a single possible answer.
- As can be noted, with the extra statistics checkbox selected we gain more data, such as the number of atoms and rules in this case.

```
% instance
motive(harry).
motive(sally).
guilty(harry).

% encoding
innocent(Suspect) :- motive(Suspect), not guilty(Suspect).
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.005s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 0.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0        (Eliminated:    0 Frozen:    0)
Constraints  : 0        (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
Consequences: [4;4]
SATISFIABLE

Models       : 1
  Brave      : yes
Consequences : 4
Calls        : 1
Time         : 0.003s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 1        (Analyzed: 0)
Restarts     : 0
Model-Level  : 0.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0        (Eliminated:    0 Frozen:    0)
Constraints  : 0        (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

PROBLEM 1 (BRAVE)

- Swapping between modes is relative in clingo. Certain problems will offer different results in different modes.
- In this case, our results are the same. However, we can take note of a few key differences in results. Namely, the consequences section.

PROBLEM 1 (CAUTIOUS & ENUMERATE ALL)

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.003s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 0.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0        (Eliminated:    0 Frozen:    0)
Constraints  : 0        (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
Consequences: [4;4]
SATISFIABLE

Models       : 1
  Cautious   : yes
Consequences : 4
Calls        : 1
Time         : 0.004s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 1        (Analyzed: 0)
Restarts     : 0
Model-Level  : 0.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0        (Eliminated:    0 Frozen:    0)
Constraints  : 0        (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

PROBLEM 1B (DEFAULT)

- Here is an example of how clingo handles complexity. How a change in a problem effects the results when it is made more complicated.
- In this case, a few notable changes are the addition of choices as explained before, through the use of curly brackets. Alongside this, there is also the large jump in rules and atoms.
- Overall, however, we can note that clingo still provides results incredibly quickly, only having gone up from 0.005s to 0.013s.

```
1  victim(tom).
2  motive(harry).
3  motive(sally).
4  witness(bob).
5  witness(tod).
6  suspect(harry).
7  suspect(sally).
8  suspect(bob).
9  suspect(tod).
10
11 {weapon(X) : motive(X)} = 1.
12 :- weapon(harry), weapon(sally).
13
14 {friends(X, Y) : motive(X)} = 1 :- witness(Y).
15 {friends(X, Y) : witness(Y)} = 1 :- motive(X).
16 {friends(X, Y) : motive(X)} = 1 :- victim(Y).
17 friends(Suspect, Victim) :- not muder(Suspect), victim(Victim), suspect(Suspect).
18
19 killed(Murder, Victim) :- muder(Murder), not friends(Murder, Victim), victim(Victim).
20 muder(Suspect) :- motive(Suspect), weapon(Suspect).
21 innocent(Suspect) :- not muder(Suspect), suspect(Suspect).
22
23 #show killed/2.
24
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
killed(harry,tom)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.013s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 2
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 2.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 64       (Original: 58)
  Choice     : 6
Atoms        : 51
Bodies       : 25
  Count      : 0        (Original: 6)
Equivalences : 36       (Atom=Atom: 8 Body=Body: 0 Other: 28)
Tight        : Yes
Variables    : 16       (Eliminated:    0 Frozen:    0)
Constraints  : 20       (Binary:  90.0% Ternary:  10.0% Other:   0.0%)
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
killed(harry,tom)
Consequences: [1;2]
Answer: 2
killed(harry,tom) killed(sally,tom)
Consequences: [2;2]
SATISFIABLE

Models       : 2
  Brave      : yes
Consequences : 2
Calls        : 1
Time         : 0.015s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 3
Conflicts    : 1        (Analyzed: 0)
Restarts     : 0
Model-Level  : 1.5
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 2        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 2        (Average Length:    0.5 Ratio: 100.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 64       (Original: 58)
  Choice     : 6
Atoms        : 51
Bodies       : 25
  Count      : 0        (Original: 6)
Equivalences : 36       (Atom=Atom: 8 Body=Body: 0 Other: 28)
Tight        : Yes
Variables    : 16       (Eliminated:    0 Frozen:    2)
Constraints  : 20       (Binary:  90.0% Ternary:  10.0% Other:   0.0%)
```

PROBLEM 1B (BRAVE)

- When swapping back to the brave mode of solving problems, we catch our first difference in terms of actual possible solutions.
- As usual, there's a substantial increase in rules, atoms and bodies. And there's another miniscule increase in time.

PROBLEM 1B (CAUTIOUS & ENUMERATE ALL)

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
killed(harry,tom)
Consequences: [0;1]
Answer: 2

Consequences: [0;0]
SATISFIABLE

Models       : 2
  Cautious   : yes
Consequences : 0
Calls        : 1
Time         : 0.010s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 5
Conflicts    : 2        (Analyzed: 1)
Restarts     : 0
Model-Level  : 1.5
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 3        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 1        (Average Length:    1.0 Ratio:  33.33%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 2        (Average Length:    0.5 Ratio:  66.67%)
Backjumps    : 1        (Average:  2.00 Max:   2 Sum:      2)
  Executed   : 1        (Average:  2.00 Max:   2 Sum:      2 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 64       (Original: 58)
  Choice     : 6
Atoms        : 51
Bodies       : 25
  Count      : 0        (Original: 6)
Equivalences : 36       (Atom=Atom: 8 Body=Body: 0 Other: 28)
Tight        : Yes
Variables    : 16       (Eliminated:    0 Frozen:    2)
Constraints  : 20       (Binary:  90.0% Ternary:  10.0% Other:   0.0%)
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
killed(harry,tom)
Answer: 2
killed(harry,tom)
Answer: 3
killed(sally,tom)
Answer: 4
killed(sally,tom)
SATISFIABLE

Models       : 4
Calls        : 1
Time         : 0.012s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 3
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 1.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 64       (Original: 58)
  Choice     : 6
Atoms        : 51
Bodies       : 25
  Count      : 0        (Original: 6)
Equivalences : 36       (Atom=Atom: 8 Body=Body: 0 Other: 28)
Tight        : Yes
Variables    : 16       (Eliminated:    0 Frozen:    0)
Constraints  : 20       (Binary:  90.0% Ternary:  10.0% Other:   0.0%)
```

PROBLEM 2 (DEFAULT)

- PROBLEM 2 IS THE CLASSIC N-QUEENS PROBLEM. THE GOAL IS TO PLACE N QUEENS ON A N X N BOARD IN SUCH A WAY SO THAT NO TWO ARE IN THE SAME ROW, COLUMN OR DIAGONAL.
- MUCH THE SAME AS OUR MODIFIED VERSION OF PROBLEM 1, THIS TIME WE'VE MADE USE OF CURLY BRACKETS TO IMPLEMENT CHOICES INTO OUR PROBLEM.
- IN THIS CASE, THE NUMBER OF RULES, ATOMS AND BODIES ARE FAR LARGER THAN EVEN THOSE IN OUR MODIFIED VERSION OF PROBLEM 1. YET, CLINGO SEEMS TO HAVE NO PROBLEM IN PROVIDING SOLUTIONS VERY QUICKLY AND ACCURATELY.

```
1  #const n = 8.
2
3  % n-Queens encoding
4
5  { q(I,1..n) } == 1 :- I = 1..n.
6  { q(1..n,J) } == 1 :- J = 1..n.
7  :- { q(D-J,J) } >= 2, D =   2..2*n.
8  :- { q(D+J,J) } >= 2, D = 1-n..n-1.
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
q(2,7) q(5,8) q(4,6) q(1,2) q(3,3) q(6,5) q(8,4) q(7,1)
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.025s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 48
Conflicts    : 15       (Analyzed: 15)
Restarts     : 0
Model-Level  : 20.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 15       (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 15       (Average Length:   13.0 Ratio: 100.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 15       (Average:  1.87 Max:   8 Sum:     28)
  Executed   : 15       (Average:  1.87 Max:   8 Sum:     28 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 380      (Original: 148)
  Choice     : 16
Atoms        : 154
Bodies       : 243      (Original: 75)
  Count      : 22       (Original: 54)
Equivalences : 112      (Atom=Atom: 16 Body=Body: 0 Other: 96)
Tight        : Yes
Variables    : 80       (Eliminated:    0 Frozen:   64)
Constraints  : 178      (Binary:  78.7% Ternary:   0.0% Other:  21.3%)
```

PROBLEM 2 (BRAVE)

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
q(2,7) q(5,8) q(4,6) q(1,2) q(3,3) q(6,5) q(8,4) q(7,1)
Consequences: [8;64]
Answer: 2
q(2,7) q(2,6) q(5,8) q(1,3) q(4,6) q(1,2) q(4,5) q(3,3) q(7,7) q(3,2) q(6,5) q(8,4) q(6,1) q(7,1)
Consequences: [14;64]
Answer: 3
q(1,7) q(2,7) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(4,5) q(7,8) q(2,2) q(3,3) q(7,7) q(3,2) q(4,3) q(6,5
Consequences: [22;64]
Answer: 4
q(1,7) q(2,7) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(7,7) q(3,2) q(4,3
Consequences: [26;64]
Answer: 5
q(1,7) q(2,7) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(7,7) q(3,2) q(4,3
Consequences: [27;64]
Answer: 6
q(1,7) q(2,8) q(2,7) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(7,7) q(3,2
Consequences: [29;64]
Answer: 7
q(1,7) q(2,8) q(2,7) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(6,6
Consequences: [33;64]
Answer: 8
q(1,7) q(2,8) q(2,7) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(6,6
Consequences: [36;64]
Answer: 9
q(1,7) q(2,8) q(2,7) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(6,6
Consequences: [37;64]
Answer: 10
q(1,7) q(2,8) q(2,7) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(7,8) q(2,2) q(3,3) q(4,4
Consequences: [38;64]
Answer: 11
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(6,7) q(7,8) q(2,2
Consequences: [41;64]
Answer: 12
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(3,6) q(5,8) q(1,3) q(4,6) q(1,2) q(3,4) q(4,5) q(6,7) q(7,8) q(2,2
Consequences: [42;64]
Answer: 13
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(3,6) q(4,7) q(5,8) q(1,3) q(4,6) q(6,8) q(1,2) q(2,3) q(3,4) q(4,5
Consequences: [48;64]
Answer: 14
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(4,6) q(6,8) q(1,2) q(2,3) q(3,4
Consequences: [49;64]
Answer: 15
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(4,6) q(6,8) q(1,2) q(2,3) q(3,4
Consequences: [50;64]
```

```
Answer: 16
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(3,5) q(4,6) q(6,8) q(1,2
Consequences: [52;64]
Answer: 17
q(1,7) q(2,8) q(2,7) q(3,8) q(1,5) q(2,6) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(3,5) q(4,6) q(6,8) q(1,2
Consequences: [53;64]
Answer: 18
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(3,5) q(4,6) q(6,8
Consequences: [55;64]
Answer: 19
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(3,5) q(4,6
Consequences: [56;64]
Answer: 20
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(2,4) q(3,5
Consequences: [58;64]
Answer: 21
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(2,4) q(3,5
Consequences: [59;64]
Answer: 22
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(3,6) q(4,7) q(5,8) q(1,3) q(2,4) q(3,5
Consequences: [60;64]
Answer: 23
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(2,5) q(3,6) q(4,7) q(5,8) q(1,3) q(2,4
Consequences: [63;64]
Answer: 24
q(1,7) q(2,8) q(1,6) q(2,7) q(3,8) q(1,5) q(2,6) q(3,7) q(4,8) q(1,4) q(2,5) q(3,6) q(4,7) q(5,8) q(1,3) q(2,4
Consequences: [64;64]
SATISFIABLE
```

```
Models       : 24
  Brave      : yes
Consequences : 64
Calls        : 1
Time         : 0.028s (Solving: 0.01s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 174
Conflicts    : 137      (Analyzed: 136)
Restarts     : 1        (Average: 136.00 Last: 115)
Model-Level  : 9.8
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 138      (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 136      (Average Length:   13.5 Ratio:  98.55%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 2        (Average Length:    0.5 Ratio:   1.45%)
Backjumps    : 136      (Average:  1.10 Max:   8 Sum:    150)
  Executed   : 136      (Average:  1.10 Max:   8 Sum:    150 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 380      (Original: 148)
  Choice     : 16
Atoms        : 154
Bodies       : 243      (Original: 75)
  Count      : 22       (Original: 54)
Equivalences : 112      (Atom=Atom: 16 Body=Body: 0 Other: 96)
Tight        : Yes
Variables    : 80       (Eliminated:    0 Frozen:   64)
Constraints  : 178      (Binary:  78.7% Ternary:   0.0% Other:  21.3%)
```

PROBLEM 2 (CAUTIOUS)

- As is the trend, clingo continues to offer efficient and swift solutions. Though, as can be seen from the brave mode of Problem 2, there are clearly cases where the results are incredibly lengthy and so may be difficult to sift through for users.
- It should be noted that both in the case of the brave and the cautious mode, the number of backjumps increases exponentially, as can be seen through the extra statistics section made available to us.

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
q(2,7) q(5,8) q(4,6) q(1,2) q(3,3) q(6,5) q(8,4) q(7,1)
Consequences: [0;8]
Answer: 2
q(5,8) q(8,4)
Consequences: [0;2]
Answer: 3

Consequences: [0;0]
SATISFIABLE

Models       : 3
  Cautious   : yes
Consequences : 0
Calls        : 1
Time         : 0.022s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 61
Conflicts    : 28       (Analyzed: 27)
Restarts     : 0
Model-Level  : 20.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 28       (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 27       (Average Length:   13.5 Ratio:  96.43%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 1        (Average Length:    0.0 Ratio:   3.57%)
Backjumps    : 27       (Average:  1.48 Max:   8 Sum:     40)
  Executed   : 27       (Average:  1.48 Max:   8 Sum:     40 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 380      (Original: 148)
  Choice     : 16
Atoms        : 154
Bodies       : 243      (Original: 75)
  Count      : 22       (Original: 54)
Equivalences : 112      (Atom=Atom: 16 Body=Body: 0 Other: 96)
Tight        : Yes
Variables    : 80       (Eliminated:    0 Frozen:   64)
Constraints  : 178      (Binary:  78.7% Ternary:   0.0% Other:  21.3%)
```

PROBLEM 2 (ENUMERATE ALL)

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
q(2,7) q(5,8) q(4,6) q(1,2) q(3,3) q(6,5) q(8,4) q(7,1)
Answer: 2
q(2,6) q(5,8) q(1,3) q(4,5) q(7,7) q(3,2) q(8,4) q(6,1)
Answer: 3
q(1,7) q(3,6) q(7,8) q(2,2) q(4,3) q(6,4) q(8,5) q(5,1)
Answer: 4
q(2,6) q(5,8) q(1,3) q(3,4) q(7,7) q(6,5) q(4,1) q(8,2)
Answer: 5
q(1,7) q(5,8) q(3,4) q(2,2) q(6,5) q(8,6) q(4,1) q(7,3)
Answer: 6
q(2,8) q(3,6) q(1,2) q(7,7) q(6,5) q(5,3) q(4,1) q(8,4)
Answer: 7
q(1,5) q(5,8) q(4,6) q(3,4) q(2,2) q(8,7) q(6,3) q(7,1)
Answer: 8
q(1,5) q(5,8) q(4,6) q(3,4) q(7,7) q(2,1) q(6,2) q(8,3)
Answer: 9
q(2,8) q(1,5) q(3,4) q(6,6) q(8,7) q(5,3) q(4,1) q(7,2)
Answer: 10
q(2,7) q(1,5) q(4,6) q(7,8) q(3,2) q(5,3) q(8,4) q(6,1)
Answer: 11
q(2,7) q(1,5) q(5,8) q(4,4) q(3,2) q(8,6) q(7,3) q(6,1)
Answer: 12
q(3,8) q(2,6) q(1,3) q(6,7) q(5,5) q(4,1) q(8,4) q(7,2)
Answer: 13
q(3,8) q(2,6) q(1,2) q(7,7) q(4,3) q(6,4) q(8,5) q(5,1)
Answer: 14
q(3,8) q(1,5) q(6,7) q(2,2) q(5,4) q(8,6) q(4,1) q(7,3)
Answer: 15
q(3,8) q(1,5) q(4,6) q(6,7) q(2,1) q(5,3) q(8,4) q(7,2)
Answer: 16
q(2,8) q(1,4) q(3,5) q(6,7) q(4,3) q(8,6) q(5,1) q(7,2)
Answer: 17
q(2,7) q(1,4) q(3,5) q(7,8) q(6,6) q(4,3) q(5,1) q(8,2)
Answer: 18
q(2,8) q(1,4) q(5,6) q(7,7) q(4,3) q(3,1) q(8,5) q(6,2)
Answer: 19
q(1,7) q(5,8) q(4,6) q(2,3) q(6,5) q(3,1) q(8,4) q(7,2)
Answer: 20
q(3,7) q(1,4) q(6,8) q(5,6) q(2,2) q(4,3) q(8,5) q(7,1)
Answer: 21
q(2,7) q(1,5) q(6,8) q(3,4) q(7,6) q(5,3) q(4,1) q(8,2)
```

```
Answer: 22
q(3,7) q(1,4) q(6,8) q(4,5) q(2,2) q(7,6) q(5,1) q(8,3)
Answer: 23
q(1,6) q(3,7) q(6,8) q(2,3) q(4,4) q(8,5) q(5,1) q(7,2)
Answer: 24
q(1,6) q(3,7) q(7,8) q(2,2) q(6,5) q(5,3) q(4,1) q(8,4)
Answer: 25
q(1,6) q(4,8) q(3,5) q(2,3) q(8,7) q(6,4) q(5,1) q(7,2)
Answer: 26
q(1,7) q(4,8) q(5,6) q(3,3) q(2,1) q(6,4) q(8,5) q(7,2)
Answer: 27
q(1,6) q(4,8) q(2,3) q(7,7) q(5,4) q(3,1) q(8,5) q(6,2)
Answer: 28
q(4,8) q(1,4) q(3,5) q(5,6) q(2,2) q(8,7) q(7,3) q(6,1)
Answer: 29
q(4,8) q(1,4) q(3,5) q(5,6) q(7,7) q(2,1) q(6,3) q(8,2)
Answer: 30
q(2,7) q(4,8) q(3,5) q(1,2) q(7,6) q(6,4) q(5,1) q(8,3)
Answer: 31
q(2,7) q(4,8) q(1,4) q(5,5) q(7,6) q(3,1) q(6,2) q(8,3)
Answer: 32
q(2,7) q(4,8) q(1,3) q(5,6) q(3,2) q(6,4) q(8,5) q(7,1)
Answer: 33
q(3,5) q(6,7) q(2,2) q(4,3) q(8,6) q(7,4) q(5,1) q(1,8)
Answer: 34
q(3,7) q(5,8) q(1,3) q(4,5) q(2,1) q(8,6) q(7,4) q(6,2)
Answer: 35
q(2,7) q(4,8) q(1,3) q(5,5) q(3,2) q(8,6) q(7,4) q(6,1)
Answer: 36
q(2,7) q(1,5) q(4,6) q(8,8) q(3,2) q(5,3) q(7,4) q(6,1)
Answer: 37
q(2,7) q(1,5) q(5,8) q(6,6) q(4,3) q(3,1) q(7,4) q(8,2)
Answer: 38
q(1,6) q(4,8) q(2,3) q(5,5) q(8,7) q(3,1) q(7,4) q(6,2)
Answer: 39
q(1,5) q(5,8) q(4,6) q(2,3) q(8,7) q(3,1) q(7,4) q(6,2)
Answer: 40
q(2,7) q(4,8) q(3,5) q(1,1) q(7,6) q(6,4) q(5,2) q(8,3)
Answer: 41
q(4,8) q(1,4) q(3,5) q(6,7) q(2,1) q(8,6) q(5,2) q(7,3)
Answer: 42
q(4,6) q(2,3) q(7,7) q(6,5) q(3,1) q(5,2) q(8,4) q(1,8)
Answer: 43
q(2,7) q(1,5) q(6,8) q(4,4) q(7,6) q(3,1) q(5,2) q(8,3)
Answer: 44
q(2,6) q(1,4) q(6,8) q(4,5) q(8,7) q(3,1) q(5,2) q(7,3)
```

```
Answer: 45
q(3,8) q(1,5) q(6,7) q(4,4) q(2,1) q(8,6) q(5,2) q(7,3)
Answer: 46
q(2,7) q(4,8) q(1,4) q(3,3) q(6,5) q(8,6) q(5,2) q(7,1)
Answer: 47
q(1,5) q(4,7) q(6,8) q(2,3) q(7,6) q(3,1) q(5,2) q(8,4)
Answer: 48
q(1,5) q(4,7) q(6,8) q(3,4) q(2,2) q(7,6) q(5,3) q(8,1)
Answer: 49
q(1,6) q(4,7) q(3,5) q(2,3) q(8,8) q(6,4) q(5,1) q(7,2)
Answer: 50
q(1,6) q(4,7) q(6,8) q(2,3) q(5,5) q(3,1) q(8,4) q(7,2)
Answer: 51
q(2,6) q(4,7) q(6,8) q(1,2) q(5,4) q(3,1) q(8,5) q(7,3)
Answer: 52
q(2,6) q(4,7) q(1,3) q(7,8) q(5,5) q(3,2) q(8,4) q(6,1)
Answer: 53
q(2,8) q(4,7) q(1,3) q(3,4) q(6,6) q(8,5) q(5,1) q(7,2)
Answer: 54
q(2,6) q(4,7) q(1,3) q(7,8) q(3,2) q(6,4) q(8,5) q(5,1)
Answer: 55
q(3,7) q(1,4) q(6,8) q(5,6) q(2,2) q(4,3) q(7,5) q(8,1)
Answer: 56
q(3,8) q(2,6) q(1,3) q(6,7) q(5,4) q(7,5) q(4,1) q(8,2)
Answer: 57
q(1,6) q(3,7) q(6,8) q(2,2) q(5,4) q(7,5) q(4,1) q(8,3)
Answer: 58
q(3,8) q(2,6) q(1,4) q(6,7) q(4,3) q(7,5) q(5,1) q(8,2)
Answer: 59
q(2,7) q(5,8) q(4,6) q(3,4) q(1,1) q(7,5) q(6,2) q(8,3)
Answer: 60
q(3,8) q(1,4) q(4,6) q(2,2) q(8,7) q(7,5) q(6,3) q(5,1)
Answer: 61
q(2,8) q(1,6) q(6,7) q(4,4) q(3,2) q(7,5) q(5,1) q(8,3)
Answer: 62
q(2,6) q(5,8) q(1,3) q(3,4) q(7,7) q(6,5) q(4,2) q(8,1)
Answer: 63
q(2,7) q(1,4) q(3,5) q(5,6) q(8,8) q(4,2) q(7,3) q(6,1)
Answer: 64
q(1,6) q(3,7) q(5,8) q(2,3) q(6,5) q(4,2) q(8,4) q(7,1)
Answer: 65
q(1,6) q(5,8) q(3,5) q(7,7) q(2,1) q(4,2) q(6,3) q(8,4)
Answer: 66
q(1,6) q(3,7) q(6,8) q(2,3) q(5,4) q(4,2) q(8,5) q(7,1)
Answer: 67
q(3,8) q(2,6) q(1,3) q(7,7) q(5,4) q(4,2) q(8,5) q(6,1)
```

```
Answer: 68
q(1,7) q(3,8) q(2,3) q(5,5) q(7,6) q(4,2) q(8,4) q(6,1)
Answer: 69
q(3,8) q(2,6) q(5,7) q(1,1) q(4,3) q(6,4) q(8,5) q(7,2)
Answer: 70
q(3,8) q(2,6) q(1,4) q(5,7) q(4,2) q(8,5) q(7,3) q(6,1)
Answer: 71
q(3,8) q(1,4) q(5,7) q(4,5) q(2,2) q(8,6) q(7,3) q(6,1)
Answer: 72
q(3,8) q(1,5) q(5,7) q(2,3) q(4,4) q(7,6) q(6,1) q(8,2)
Answer: 73
q(5,7) q(3,4) q(2,2) q(6,5) q(8,6) q(4,1) q(7,3) q(1,8)
Answer: 74
q(2,8) q(1,5) q(5,7) q(3,4) q(7,6) q(4,1) q(6,2) q(8,3)
Answer: 75
q(1,5) q(3,6) q(5,7) q(7,8) q(2,2) q(6,4) q(4,1) q(8,3)
Answer: 76
q(2,8) q(1,4) q(5,7) q(4,5) q(7,6) q(3,1) q(6,2) q(8,3)
Answer: 77
q(1,6) q(3,7) q(5,8) q(2,4) q(7,5) q(4,1) q(6,2) q(8,3)
Answer: 78
q(1,6) q(3,7) q(2,4) q(8,8) q(6,5) q(5,3) q(4,1) q(7,2)
Answer: 79
q(4,8) q(3,6) q(2,4) q(1,2) q(7,7) q(5,3) q(8,5) q(6,1)
Answer: 80
q(1,6) q(4,8) q(2,4) q(6,7) q(5,5) q(3,2) q(8,3) q(7,1)
Answer: 81
q(1,6) q(5,8) q(2,4) q(4,5) q(7,7) q(3,1) q(6,2) q(8,3)
Answer: 82
q(1,7) q(5,8) q(2,4) q(4,5) q(3,2) q(8,6) q(7,3) q(6,1)
Answer: 83
q(1,7) q(4,8) q(2,4) q(5,6) q(3,2) q(8,5) q(7,3) q(6,1)
Answer: 84
q(2,4) q(5,6) q(7,7) q(4,3) q(3,1) q(8,5) q(6,2) q(1,8)
Answer: 85
q(3,7) q(2,5) q(6,8) q(1,2) q(4,4) q(7,6) q(5,1) q(8,3)
Answer: 86
q(3,7) q(2,5) q(1,3) q(7,8) q(5,4) q(8,6) q(4,1) q(6,2)
Answer: 87
q(1,7) q(2,5) q(6,8) q(5,6) q(3,3) q(4,1) q(8,4) q(7,2)
Answer: 88
q(3,7) q(2,5) q(6,8) q(1,2) q(7,6) q(5,3) q(4,1) q(8,4)
Answer: 89
q(3,8) q(2,5) q(4,6) q(6,7) q(1,1) q(5,3) q(8,4) q(7,2)
Answer: 90
q(4,8) q(2,5) q(1,3) q(5,6) q(7,7) q(3,2) q(6,4) q(8,1)
Answer: 91
q(3,8) q(2,5) q(1,3) q(6,7) q(4,4) q(8,6) q(5,1) q(7,2)
```

PROBLEM 2 (ENUMERATE ALL) CONTINUED

- As mentioned, some modes provide a rather large set of possibilities. Making it particularly difficult for users to trace exact details regarding problems such as these that seem to have a spectacularly large answer set.

```
Answer: 89
q(3,8) q(2,5) q(4,6) q(6,7) q(1,1) q(5,3) q(8,4) q(7,2)
Answer: 90
q(4,8) q(2,5) q(1,3) q(5,6) q(7,7) q(3,2) q(6,4) q(8,1)
Answer: 91
q(3,8) q(2,5) q(1,3) q(6,7) q(4,4) q(8,6) q(5,1) q(7,2)
Answer: 92
q(4,8) q(2,5) q(1,3) q(6,7) q(3,2) q(8,6) q(7,4) q(5,1)
SATISFIABLE

Models       : 92
Calls        : 1
Time         : 0.032s (Solving: 0.01s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 446
Conflicts    : 336      (Analyzed: 280)
Restarts     : 14       (Average: 20.00 Last: 257)
Model-Level  : 8.1
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 280      (Deleted: 0)
  Binary     : 3        (Ratio:   1.07%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 280      (Average Length:   13.0 Ratio: 100.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 280      (Average:  1.07 Max:   8 Sum:    299)
  Executed   : 280      (Average:  1.07 Max:   8 Sum:    299 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 380      (Original: 148)
  Choice     : 16
Atoms        : 154
Bodies       : 243      (Original: 75)
  Count      : 22       (Original: 54)
Equivalences : 112      (Atom=Atom: 16 Body=Body: 0 Other: 96)
Tight        : Yes
Variables    : 80       (Eliminated:    0 Frozen:   64)
Constraints  : 178      (Binary:  78.7% Ternary:   0.0% Other:  21.3%)
```

PROBLEM 3 (DEFAULT)

- PROBLEM 3 IS A PROBLEM BASED UPON THE FOUR-COLOR THEOREM, WHICH STATES THAT ANY MAP IN A PLANE CAN BE COLORED USING FOUR COLORS IN SUCH A MANNER, THAT NO TWO AREAS THAT SHARE A COMMON BOUNDARY WILL SHARE THE SAME COLOR.
- IN THIS PROBLEM, WE'VE EXPLICITLY DEFINED MANY EDGES IN ORDER TO SIMULATE THIS THEOREM.

```
% Default
#const c = 4.
#const n = 48.
% Generate
{ color(X,1..c) } = 1 :- node(X).
% Test
:- edge(X,Y), color(X,C), color(Y,C).
% Nodes
node(1..n).
% (Directed) Edges
edge(1,(2;7)). edge(2,(1;5;3;7)). edge(3,(2;5;4)). edge(5,(2;3;7;6;4)).
edge(7,(1;2;5;8;9;6)). edge(8,(7;9;16;17)). edge(9,(7;8;6;10;15;16)).
edge(6,(5;7;9;4;10)). edge(4,(3;5;6;11)). edge(11,(4;10;13;12)).
edge(10,(9;6;11;15;14;13)). edge(15,(9;10;16;19;20;14)). edge(16,(8;9;15;17;18;19)).
edge(17,(8;16;18)). edge(18,(16;17;19;23)). edge(19,(15;16;18;20;25;23)).
edge(20,(15;19;14;13;21;29;25)). edge(14,(10;15;20;13)). edge(13,(11;10;14;20;12;21)).
edge(12,(11;13;22;21)). edge(22,(12;21;30)). edge(21,(20;13;12;22;30;29)). edge(30,(22;21;31;29)).
edge(31,(30;33;32;29)). edge(33,(31;32)). edge(32,(31;33;34;29;35)). edge(34,(32;35)).
edge(29,(20;21;30;31;32;34;35;28;36)). edge(35,(32;34;29;36)). edge(28,(20;29;25;26;27;37;36)).
edge(25,(19;20;12;23;26)). edge(23,(18;19;25)). edge(24,(26;27)). edge(26,(28;25;24;27)).
edge(27,(28;24;26;37;38)). edge(37,(28;27;36;48;38)). edge(36,(29;35;28;37;48)).
edge(48,(37;36;47;38)). edge(47,(48;46;38)). edge(46,(47;38;39)). edge(38,(27,37,48,47,46,39)).
edge(39,(46;38;40;43;45)). edge(40,(39;41;43)). edge(42,(41)). edge(41,(40;42;43)).
edge(43,(39;40;41;44;45)). edge(44,(43;45)). edge(45,(39;43;44)).
% Display
#show color/2.
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
color(2,1) color(7,2) color(1,4) color(3,2) color(5,4) color(4,3) color(6,1) color(11,4) color(10,3) col
SATISFIABLE

Models       : 1+
Calls        : 1
Time         : 0.060s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 42
Conflicts    : 0        (Analyzed: 0)
Restarts     : 0
Model-Level  : 42.0
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 0        (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 0        (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0        (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0        (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 1731     (Original: 1347)
  Choice     : 48
Atoms        : 635
Bodies       : 953      (Original: 569)
  Count      : 0        (Original: 96)
Equivalences : 336      (Atom=Atom: 48 Body=Body: 0 Other: 288)
Tight        : Yes
Variables    : 240      (Eliminated:    0 Frozen:    0)
Constraints  : 760      (Binary:  93.7% Ternary:   0.0% Other:   6.3%)
```

PROBLEM 3 (BRAVE & CAUTIOUS)

```
Models       : 72
  Brave      : yes
Consequences : 192
Calls        : 1
Time         : 0.072s (Solving: 0.01s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 1329
Conflicts    : 48       (Analyzed: 47)
Restarts     : 0
Model-Level  : 35.1
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 49       (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 1        (Ratio:   2.04%)
  Conflict   : 47       (Average Length:   14.4 Ratio:  95.92%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 2        (Average Length:    0.5 Ratio:   4.08%)
Backjumps    : 47       (Average:  1.43 Max:   4 Sum:     67)
  Executed   : 47       (Average:  1.43 Max:   4 Sum:     67 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 1731     (Original: 1347)
  Choice     : 48
Atoms        : 635
Bodies       : 953      (Original: 569)
  Count      : 0        (Original: 96)
Equivalences : 336      (Atom=Atom: 48 Body=Body: 0 Other: 288)
Tight        : Yes
Variables    : 240      (Eliminated:    0 Frozen:  192)
Constraints  : 760      (Binary:  93.7% Ternary:   0.0% Other:   6.3%)
```

```
Models       : 28
  Cautious   : yes
Consequences : 0
Calls        : 1
Time         : 0.063s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 601
Conflicts    : 13       (Analyzed: 12)
Restarts     : 0
Model-Level  : 44.6
Problems     : 1        (Average Length: 0.00 Splits: 0)
Lemmas       : 14       (Deleted: 0)
  Binary     : 0        (Ratio:   0.00%)
  Ternary    : 0        (Ratio:   0.00%)
  Conflict   : 12       (Average Length:    9.3 Ratio:  85.71%)
  Loop       : 0        (Average Length:    0.0 Ratio:   0.00%)
  Other      : 2        (Average Length:    0.5 Ratio:  14.29%)
Backjumps    : 12       (Average:  1.75 Max:   4 Sum:     21)
  Executed   : 12       (Average:  1.75 Max:   4 Sum:     21 Ratio: 100.00%)
  Bounded    : 0        (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)

Rules        : 1731     (Original: 1347)
  Choice     : 48
Atoms        : 635
Bodies       : 953      (Original: 569)
  Count      : 0        (Original: 96)
Equivalences : 336      (Atom=Atom: 48 Body=Body: 0 Other: 288)
Tight        : Yes
Variables    : 240      (Eliminated:    0 Frozen:  192)
Constraints  : 760      (Binary:  93.7% Ternary:   0.0% Other:   6.3%)
```

Problem 3 (Enumerate All)

- This was one of the unique cases that we came across, where clingo crashed. This tells us that there are specific cases where clingo can not handle.
- The most likely case for this crash was the sheer number of cases through which the software needed to run through, leading to it crashing due to exceeding some internal resource quota which was allocated to it.

SMODELS + LPARSE
AS PRESENTED BY: Arya Mazloomi

- The program smodels is an implementation of the stable model semantics for logic programs.
- Lparse is a front-end for smodels that generates a variable-free simple logic program that can be given to smodels.

```
// Compute the stable models of the program
//    a :- not b.
//    b :- not a.
#include <iostream>
#include "smodels.h"
#include "api.h"
#include "atomrule.h"

using namespace std;

int main ()
{
  Smodels smodels;
  Api api (&smodels.program);

  // You'll have to keep track of the atoms not remembered yourself
  api.remember ();

  Atom *a = api.new_atom ();
  Atom *b = api.new_atom ();
  api.set_name (a, "a");      // You can give the atoms names.
  api.set_name (b, "b");

  api.begin_rule (BASICRULE);
  api.add_head (a);
  api.add_body (b, false);  // Add "not b" to the body.
  api.end_rule ();
  api.begin_rule (BASICRULE);
  api.add_head (b);
  api.add_body (a, false);
  api.end_rule ();

  // You would add the compute statement here, e.g.,
  // api.set_compute (a, true) demands that a is in the model.

  api.done ();  // After this you shouldn't change the rules.

  smodels.program.print ();  // You can display the program.

  smodels.init ();  // Must be called before computing any models.

  // Compute all stable models.
  while (smodels.model ())  // Returns 0 when there are no more models
    smodels.printAnswer ();  // Prints the answer

  // Of course, you can inspect the atoms directly.

  smodels.revert (true);  // Forget everything that happened after init ().

  b->computeFalse = true;  // compute { not b }
  // Alternatively, api.set_compute (b, false).
  // api.reset_compute (Atom *, bool) removes atoms from the compute
  // statement.

  if (smodels.model ())  // There is a model.
    {
      Atom *c = api.get_atom ("a");
      if (c->Bpos)
        cout << c->atom_name () << " is in the stable model" << endl;
      if (c->Bneg)
        cout << c->atom_name () << " is not in the stable model" << endl;
    }

  return 0;
}
```

SModels → LParse

```
a :- not b.
b :- not a.
compute {not b}.
```

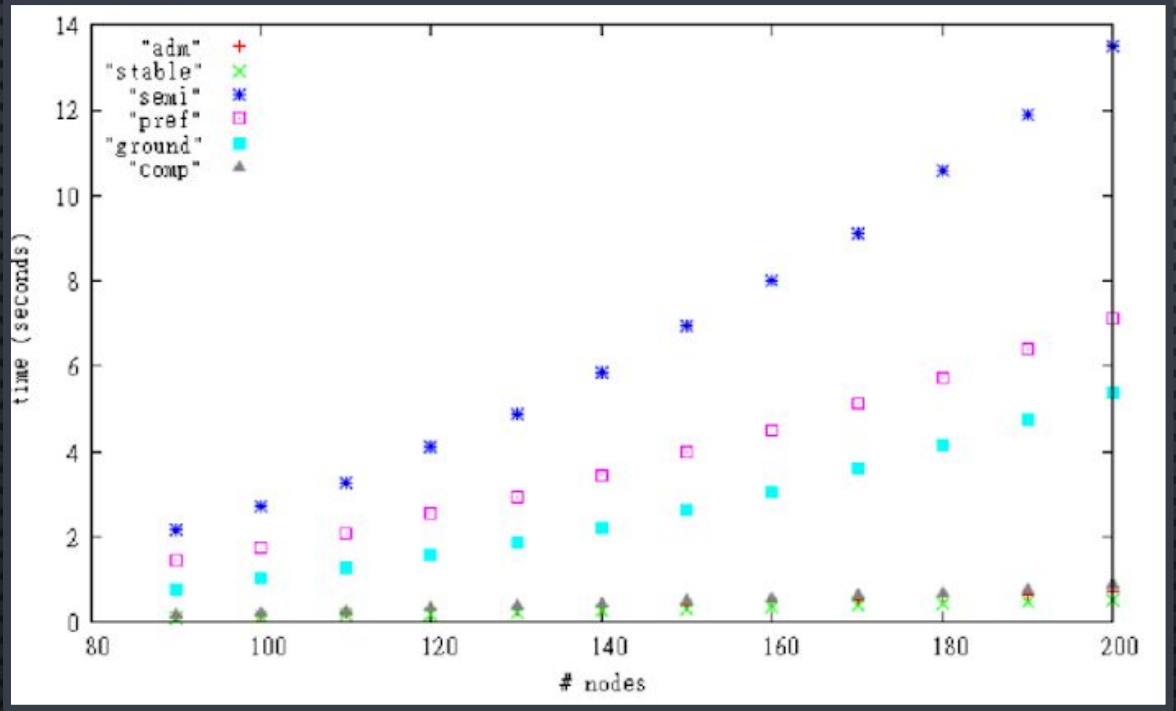
CLINGO VS SMODELS

```
motive(harry).
motive(sally).
guilty(harry).


innocent(Suspect) :- motive(Suspect), not guilty(Suspect).
```

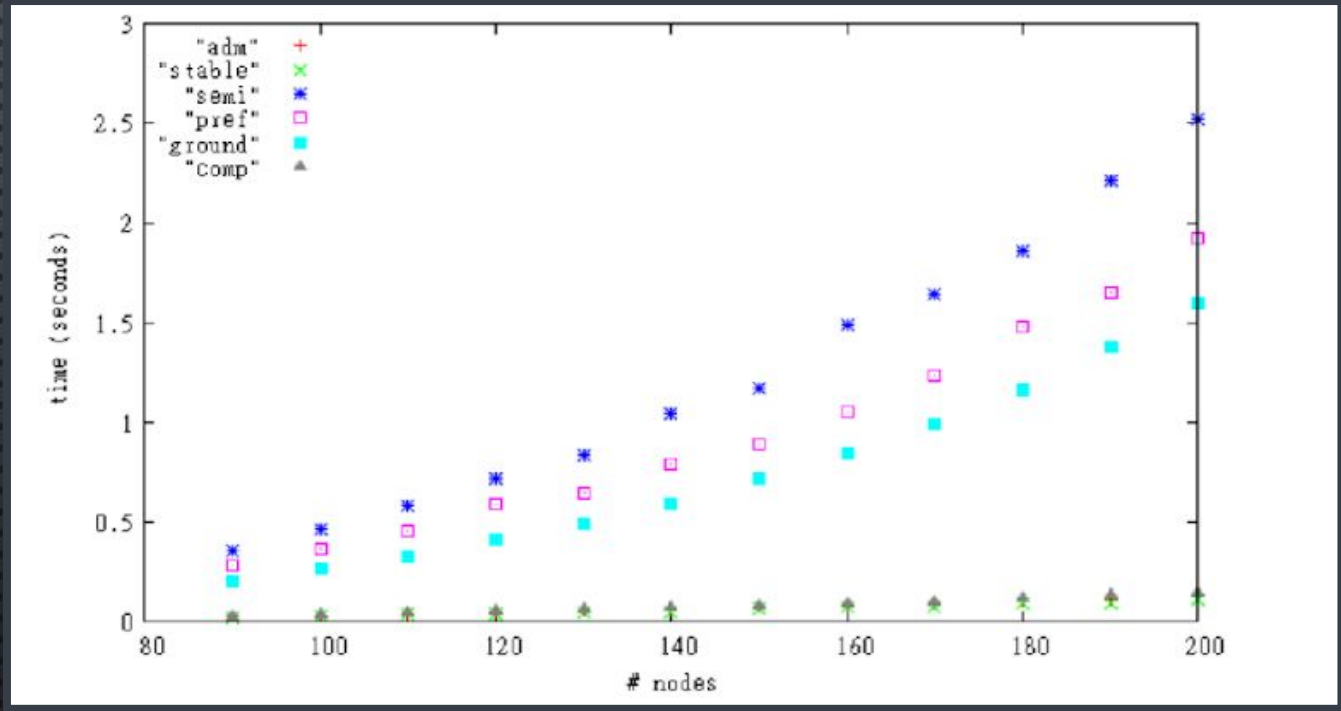
```
adminadmin@adminadmin-virtual-machine:~/Desktop/lparse-1.1.2/examples/logic$ lparse h1.lp | smodels
smodels version 2.34. Reading...done
Answer: 1
Stable Model: innocent(sally) guilty(harry) motive(sally) motive(harry)
True
Duration: 0.0
Number of choice points: 0
Number of wrong choices: 0
Number of atoms: 5
Number of rules: 4
Number of picked atoms: 0
Number of forced atoms: 0
Number of truth assignments: 4
Size of searchspace (removed): 0 (0)
adminadmin@adminadmin-virtual-machine:~/Desktop/lparse-1.1.2/examples/logic$
```

```
clingo version 5.5.0
Reading from stdin
Solving...
Answer: 1
motive(harry) motive(sally) guilty(harry) innocent(sally)
SATISFIABLE

Models       : 1
Calls        : 1
Time         : 0.000s (Solving: 0.00s 1st Model: 0.00s Unsat: 0.00s)
CPU Time     : 0.000s

Choices      : 0
Conflicts    : 0          (Analyzed: 0)
Restarts     : 0
Model Level  : 0.0
Problems     : 1          (Average Length: 0.00 Splits: 0)
Lemmas       : 0          (Deleted: 0)
  Binary     : 0          (Ratio:    0.00%)
  Ternary    : 0          (Ratio:    0.00%)
  Conflict   : 0          (Average Length:    0.0 Ratio:   0.00%)
  Loop       : 0          (Average Length:    0.0 Ratio:   0.00%)
  Other      : 0          (Average Length:    0.0 Ratio:   0.00%)
Backjumps    : 0          (Average:  0.00 Max:   0 Sum:      0)
  Executed   : 0          (Average:  0.00 Max:   0 Sum:      0 Ratio:   0.00%)
  Bounded    : 0          (Average:  0.00 Max:   0 Sum:      0 Ratio: 100.00%)

Rules        : 4
Atoms        : 4
Bodies       : 0
Tight        : Yes
Variables    : 0          (Eliminated:    0 Frozen:    0)
Constraints  : 0          (Binary:   0.0% Ternary:   0.0% Other:   0.0%)
```

Lparse + SModels



Clingo



Thank you for your time.
We will be taking any questions you have now.