KNOWLEDGE REPRESENTATION, REASONING AND DECLARATIVE PROBLEM SOLVING

by Jicheng Zhao

September 24: Declarative Problem Solving: N-queens problem

N-queens problem declaration

- Problem: We have a $n \times n$ board. We need to place n queens onto the board so that no two queens attack each other. Two queens attack each other if they are in the same row, the same column, or along the same diagonal line.
- Problem declaration:

$$\begin{aligned} queen(1),\ \ldots,\ queen(n). \\ row(1),\ \ldots,\ row(n). \\ col(1),\ \ldots,\ col(n). \end{aligned}$$

- Conditions to be satisfied:
  1. For each location and each queen, either the queen is placed on the location or not.
  2. Every queen must be in some location.

3. A queen cannot be in more than one location.
4. At most one queen is placed on one location.
5. No two queens can be in the same row.
6. No two queens can be in the same column.
7. No two queens can be along the same diagonal.

This is the exact definition of the problem. Now, we exam the method of encoding the problem in $AnsProlog$.

An intuitive encoding of the problem

The direction encoding is got by translating all the constraints mentioned above into $AnsProlog$.

1. For each location and each queen, either the queen is placed on the location or not.

$$\begin{aligned} at(I,X,Y) &\leftarrow queen(I), row(X), col(Y), not\ not\_at(I,X,Y). \\ not\_at(I,X,Y) &\leftarrow queen(I), row(X), col(Y), not\ at(I,X,Y). \end{aligned}$$

2. Every queen must be in some location.

$$\begin{aligned} placed(I) &\leftarrow queen(I), row(X), col(Y), at(I,X,Y). \\ &\leftarrow queen(I), not\ placed(I). \end{aligned} \tag{1}$$

3. A queen cannot be in more than one location.

$$\leftarrow queen(I), row(X), col(Y), row(U), col(Z), at(I,X,Y), at(I,U,Z), Y \neq Z.$$
$$\leftarrow queen(I), row(X), col(Y), row(U), col(Z), at(I,X,Y), at(I,U,Z), X \neq U.$$

4. At most one queen is placed on one location.

$$\leftarrow queen(I), queen(J), row(X), col(Y), at(I,X,Y), at(J,X,Y), I \neq J.$$

5. No two queens can be in the same row.

$$\leftarrow queen(I), row(X), col(Y), col(V), queen(J), at(I,X,Y), at(J,X,V), I \neq J.$$

6. No two queens can be in the same column.

$$\leftarrow queen(I), row(X), col(Y), row(V), queen(J), at(I,X,Y), at(J,V,Y), I \neq J.$$

7. No two queens can be along the same diagonal.

$$\leftarrow queen(I), row(X), col(Y), queen(J), row(Z), col(W), at(I,X,Y), at(J,Z,W), I \neq J, abs(X-Z) = abs(Y-W).$$

Some predicates needed in the program are defined as:

- $at(I, X, Y)$: queen I is in location (X, Y).
- $not\_at(I, X, Y)$: queen I is not in location (X, Y).
- $placed(I)$: the queen I is placed in somewhere of the board.

In $abs(X - Z) = abs(Y - W)$, the $abs(X)$ means the absolute value of integer X.

Improve the encoding by the technique of the encoding of "exactly one"

1. Item (**??**) subsumes Item (**??**). So Item (**??**) can be removed from the program without affecting the meaning of the program.
2. The following item can replace Item (**??**), Item (**??**), and Item (**??**).
   - Each queen must be placed in a unique location.

   This is encoded as:

$$\begin{aligned}
other_at(I,X,Y) &\leftarrow queen(I), row(X), col(Y), row(U), col(Z), at(I,U,Z), Y \neq Z.\\
other_at(I,X,Y) &\leftarrow queen(I), row(X), col(Y), row(U), col(Z), at(I,U,Z), X \neq U.\\
at(I,X,Y) &\leftarrow queen(I), row(X), col(Y), not\ other_at(I,X,Y)
\end{aligned}$$

3. The "exactly one" also be used to define "there is exactly one queen in each column (row, diagonal)". Accordingly, the rule can be:

$$other\_at(I,X,Y) \leftarrow queen(I), row(X), col(Y), col(V), queen(J), at(J,X,V), I \neq J.$$
$$other\_at(I,X,Y) \leftarrow queen(I), row(X), col(Y), row(U), queen(J), at(J,U,Y), I \neq J.$$
$$other\_at(I,X,Y) \leftarrow queen(I), row(X), col(Y), col(V), row(U), queen(J), at(J,V,U), I \neq J, abs(X-U) = abs(Y-V).$$

Further Simplify with the candidate constraint program

- In the previous pages, we intended to present the meaning of "exactly one". This can be easily handled by the candidate constraint programs (Also known as weighted logic programs).

- Short introduction to the candidate constraint programs.

  We can write the constraint that each queen appears in one and only one location by

$$1\{at(I,X,Y) : row(X), col(Y)\}1 \leftarrow queen(I).$$

  Is a shorthand for the following n rules:

$$\begin{aligned} 1\{at(I,X,Y) : row(X), row(Y)\}1 \quad &\leftarrow \quad queen(1). \\ &\ldots \\ 1\{at(I,X,Y) : row(X), row(Y)\}1 \; &\leftarrow queen(n). \end{aligned}$$

  Where each rule

$$1\{at(I,X,Y) : row(X), col(Y)\}1 \leftarrow queen(1).$$

  means that if $queen(1)$ is true is in the program, then after grounding "row(X) : col(Y)" as following:

$$1\{at(1,1,1), at(1,1,2), at(1,1,3), \ldots, at(1,n,n)\}1 \leftarrow queen(1).$$

exactly one atom in the head can be true.

- With the candidate constraint program, the N-queens problem can be encoded as:

$$\begin{aligned}
1\{at(I,X,Y) : row(X), col(Y)\}1 &\leftarrow queen(I).\\
1\{at(I,X,Y) : row(X), queen(I)\}1 &\leftarrow row(X).\\
1\{at(I,X,Y) : col(Y), queen(I)\}1 &\leftarrow col(Y).\\
&\leftarrow at(I,X,Y), at(J,U,V), queen(I), queen(J), row(X), col(Y), row(U), col(V), abs(X-U) = abs(Y-V).
\end{aligned}$$

Simplify with different domain formulation

- There is no need to distinguish different queens. That is, instead of using $at(I, X, Y)$ to denote a queen in location (X, Y), we use $at(X, Y)$ to denote the same meaning.
- The encoding can be:

$$
\begin{aligned}
other\_at(X,Y) &\leftarrow row(X), col(Y), col(Z), at(X,Z), Y \neq Z.\\
other\_at(X,Y) &\leftarrow row(X), col(Y), row(U), at(U,Y), X \neq U.\\
other\_at(X,Y) &\leftarrow row(X), col(Y), row(U), col(Z), at(U,Z), abs(X-U) = abs(Y-V), X \neq U.\\
at(X,Y) &\leftarrow row(X), col(Y), not\ other\_at(X,Y).
\end{aligned}
$$

- Besides, we also need to define that all queens are placed on the board. In the other word, the n queens are all placed on the $n \times n$ board.

$$\begin{aligned} has\_queen(X) &\leftarrow row(X), col(Y), at(X,Y). \\ &\leftarrow row(X), not\ has\_queen(X). \end{aligned}$$

  where $has\_queen(X)$ means that there is a queen in row X.

- We expect that the search space of this encoding is reduced.