Handout 6

Methodology of Answer Set Programming

Recall that solving a problem using ASP means to write a logic program whose answer sets correspond to solutions, and then find solutions using an answer set solver. The basic approach to writing such programs consists in combining choice rules with constraints (Handout 5). In this handout we discuss a few examples of this "generate-and-test" strategy.

Example: N Queens

Our goal is to place n queens on an $n \times n$ chessboard so that no two queens would be placed on the same row, column or diagonal. A solution can be described by a set of atoms of the form $q(i, j)$ $(1 \leq i, j \leq n)$; including $q(i, j)$ in the set indicates that there is a queen at position (i, j). A solution is a set X satisfying the following conditions:

1. The cardinality of X is n.
2. X does not contain a pair of different atoms of the form $q(i, j)$, $q(i', j)$ (two queens on the same column).
3. X does not contain a pair of different atoms of the form $q(i, j)$, $q(i, j')$ (two queens on the same row).
4. X does not contain a pair of different atoms $q(i, j)$, $q(i', j')$ with $|i' - i| = |j' - j|$ (two queens on the same diagonal).

The sets satisfying Conditions 1 and 2 can be described by rules similar to the program from Problem 5.9:

$$1 \leq \{q(1, j), \ldots, q(n, j)\}^c \leq 1 \qquad (1 \leq j \leq n)$$

(exactly one queen on each column). These rules form the "generate" part of our program. The "test" part consists of the constraints expressing Condition 3

$$\leftarrow q(i, j), q(i, j') \qquad (1 \leq i, j, j' \leq n;\ j < j')$$

and Condition 4

$$\leftarrow q(i, j), q(i', j') \qquad (1 \leq i, i' j, j' \leq n;\ j < j';\ |i' - i| = j' - j).$$

Here is a representation of this program in the input language of CLINGO:

```
number(1..n).
#domain number(I;I1;J;J1).

1{q(K,J) : number(K)}1.

:- q(I,J), q(I,J1), J<J1.

:- q(I,J), q(I1,J1), J<J1, abs(I1-I)==J1-J.
```

The expression `number(I;I1;J;J1)` is the CLINGO abbreviation for the list

```
number(I), number(I1), number(J), number(J1).
```

6.1 Use CLINGO to find all solutions to the 8 queens problem that (a) have a queen at (1,1); (b) have no queens in the 4×4 square in the middle of the board.

6.2 (a) Check how long it takes for CLINGO to find one solution to the 16 queens problem using the program above. (b) The constraint corresponding to Condition 3 can be alternatively written using a cardinality expression:

$$\leftarrow 2 \leq \{q(i,1), \ldots, q(i,n)\} \qquad (1 \leq i \leq n).$$

Modify the program accordingly and check how this modification affects the computation time for 16 queens.

Example: Schur Numbers

We will show now how to use ASP to estimate the size of Schur numbers. A set A of integers is called *sum-free* if there are no numbers x, y in A such that $x + y$ is in A also (x and y do not need to be different). The *Schur number* $S(k)$ is the largest integer n for which the interval $\{1, \ldots, n\}$ can be partitioned into k sum-free sets.[1]

For specific values of k, $S(k)$ can be computed (or estimated) using a program whose answer sets correspond to the partitions of $\{1, \ldots, n\}$ into k (possibly empty) sum-free sets. We will use the atoms $a_i(x)$ ($1 \leq i \leq k$, $1 \leq x \leq n$) to express that x belongs to the i-th set A_i. The choice rules

$$1 \leq \{a_1(x), \ldots, a_k(x)\} \leq 1 \qquad (1 \leq x \leq n)$$

describe "potential solutions"—partitions of $\{1, \ldots, n\}$ into k sets. The constraints

$$\leftarrow a_i(x), a_i(y), a_i(x+y) \qquad (x, y \geq 1,\ x + y \leq n,\ 1 \leq i \leq k)$$

[1] See `http://mathworld.wolfram.com/SchurNumber.html` .

eliminate the sets A_i that are not sum-free.

Here is how this program can be written in the input language of CLINGO:

```
number(1..n).
#domain number(X;Y).

subset(1..k).

1{a(I,X) : subset(I)}1.

:- a(I,X), a(I,Y), a(I,X+Y), subset(I), X+Y<=n.
```

6.3 Use this program to find $S(3)$.

Programming Projects

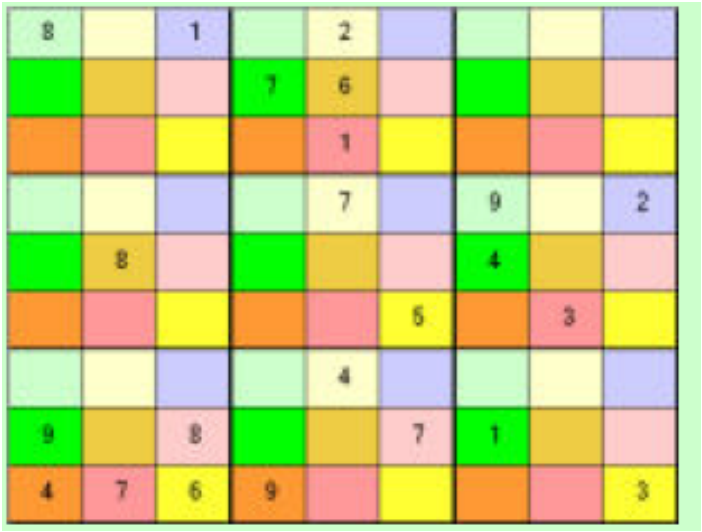
In each of the following exercises, show how to solve the given computational problem using CLINGO.

6.4 Sudoku (a.k.a. Number Place) is a game to fill in the grid so that every row, every column, and every 3×3 box contains all the digits 1 through 9.

	6		1		4		5	
		8	3		5	6		
2								1
8			4		7			6
		6				3		
7			9		1			4
5								2
		7	2		6	9		
	4		5		8		7	

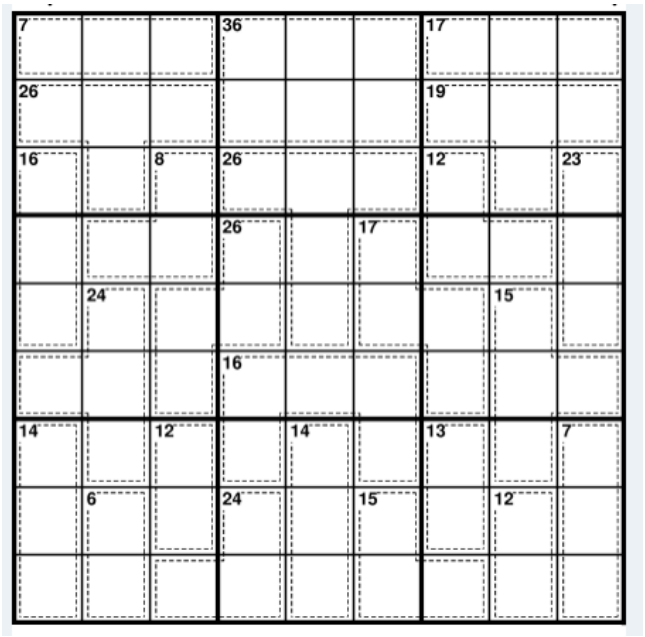
Problem: fill in the numbers.

6.5 Consider the following variant of Sudoku. The same position in each of 3×3 boxes form a "region," yielding 9 total regions (a region is represented by the same color below). In addition to the requirement of Sudoku, every region must contain all the digits 1 through 9.



Problem: fill in the numbers.

6.6 Consider the following variant of Sudoku. Each cage ("dotted area") is associated with a number. In addition to the requirement of Sudoku, the sum of the cells in a cage must be equal to the number given for the cage. Each digit in the cage must be unique.



Problem: fill in the numbers.