Global constraints involve an arbitrary number of variables (but not necessarily all)

**Example (All different)**

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|
| A | | | 3 | | 2 | | 6 | | |
| B | 9 | | | 3 | | 5 | | | 1 |
| C | | | 1 | 8 | | 6 | 4 | | |
| D | | | 8 | 1 | | 2 | 9 | | |
| E | 7 | | | | | | | | 8 |
| F | | | 6 | 7 | | 8 | 2 | | |
| G | | | 2 | 6 | | 9 | 5 | | |
| H | 8 | | | 2 | | 3 | | | 9 |
| I | | | 5 | | 1 | | 3 | | |

Alldiff (A1,A2,A3,A4,A5,A6, A7, A8, A9)
Alldiff (B1,B2,B3,B4,B5,B6,B7,B8,B9)

...

Alldiff (A1,B1,C1,D1,E1, F1,G1,H1, I1)
Alldiff (A2,B2,C2,D2,E2, F2,G2,H2, I2)

...

Alldiff (A1,A2,A3,B1,B2,B3,C1,C2,C3)
Alldiff (A4,A5,A6,B4,B5,B6,C4,C5,C6)

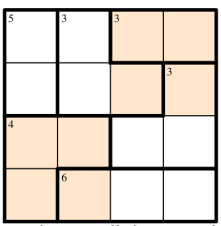
A logician taking ginkgo biloba

```
assign(domain_size,9).
assign(max_models,-1).

list(distinct).
  [f(0,0),f(0,1),f(0,2),f(0,3),f(0,4),f(0,5),f(0,6),f(0,7),f(0,8)].
  [f(1,0),f(1,1),f(1,2),f(1,3),f(1,4),f(1,5),f(1,6),f(1,7),f(1,8)].
  [f(2,0),f(2,1),f(2,2),f(2,3),f(2,4),f(2,5),f(2,6),f(2,7),f(2,8)].
  [f(3,0),f(3,1),f(3,2),f(3,3),f(3,4),f(3,5),f(3,6),f(3,7),f(3,8)].
  [f(4,0),f(4,1),f(4,2),f(4,3),f(4,4),f(4,5),f(4,6),f(4,7),f(4,8)].
  [f(5,0),f(5,1),f(5,2),f(5,3),f(5,4),f(5,5),f(5,6),f(5,7),f(5,8)].
  [f(6,0),f(6,1),f(6,2),f(6,3),f(6,4),f(6,5),f(6,6),f(6,7),f(6,8)].
  [f(7,0),f(7,1),f(7,2),f(7,3),f(7,4),f(7,5),f(7,6),f(7,7),f(7,8)].
  [f(8,0),f(8,1),f(8,2),f(8,3),f(8,4),f(8,5),f(8,6),f(8,7),f(8,8)].

  [f(0,0),f(1,0),f(2,0),f(3,0),f(4,0),f(5,0),f(6,0),f(7,0),f(8,0)].
  [f(0,1),f(1,1),f(2,1),f(3,1),f(4,1),f(5,1),f(6,1),f(7,1),f(8,1)].
  [f(0,2),f(1,2),f(2,2),f(3,2),f(4,2),f(5,2),f(6,2),f(7,2),f(8,2)].
  [f(0,3),f(1,3),f(2,3),f(3,3),f(4,3),f(5,3),f(6,3),f(7,3),f(8,3)].
  [f(0,4),f(1,4),f(2,4),f(3,4),f(4,4),f(5,4),f(6,4),f(7,4),f(8,4)].
  [f(0,5),f(1,5),f(2,5),f(3,5),f(4,5),f(5,5),f(6,5),f(7,5),f(8,5)].
  [f(0,6),f(1,6),f(2,6),f(3,6),f(4,6),f(5,6),f(6,6),f(7,6),f(8,6)].
  [f(0,7),f(1,7),f(2,7),f(3,7),f(4,7),f(5,7),f(6,7),f(7,7),f(8,7)].
  [f(0,8),f(1,8),f(2,8),f(3,8),f(4,8),f(5,8),f(6,8),f(7,8),f(8,8)].

  [f(0,0),f(0,1),f(0,2),f(1,0),f(1,1),f(1,2),f(2,0),f(2,1),f(2,2)].
  [f(3,0),f(3,1),f(3,2),f(4,0),f(4,1),f(4,2),f(5,0),f(5,1),f(5,2)].
  [f(6,0),f(6,1),f(6,2),f(7,0),f(7,1),f(7,2),f(8,0),f(8,1),f(8,2)].
  [f(0,3),f(0,4),f(0,5),f(1,3),f(1,4),f(1,5),f(2,3),f(2,4),f(2,5)].
  [f(0,6),f(0,7),f(0,8),f(1,6),f(1,7),f(1,8),f(2,6),f(2,7),f(2,8)].
  [f(3,3),f(3,4),f(3,5),f(4,3),f(4,4),f(4,5),f(5,3),f(5,4),f(5,5)].
  [f(3,6),f(3,7),f(3,8),f(4,6),f(4,7),f(4,8),f(5,6),f(5,7),f(5,8)].
  [f(6,3),f(6,4),f(6,5),f(7,3),f(7,4),f(7,5),f(8,3),f(8,4),f(8,5)].
  [f(6,6),f(6,7),f(6,8),f(7,6),f(7,7),f(7,8),f(8,6),f(8,7),f(8,8)].
end_of_list.

formulas(puzzle).
  f(0,2)=2.  f(0,4)=1.  f(0,6)=5.
  f(1,0)=8.  f(1,3)=2.  f(1,5)=4.   f(1,8)=0.
  f(2,2)=0.  f(2,3)=7.  f(2,5)=5.   f(2,6)=3.
  f(3,2)=7.  f(3,3)=0.  f(3,5)=1.   f(3,6)=8.
  f(4,0)=6.  f(4,8)=7.
  f(5,2)=5.  f(5,3)=6.  f(5,5)=7.   f(5,6)=1.
  f(6,2)=1.  f(6,3)=5.  f(6,5)=8.   f(6,6)=4.
  f(7,0)=7.  f(7,3)=1.  f(7,5)=2.   f(7,8)=8.
  f(8,2)=4.  f(8,4)=0.  f(8,6)=2.
end_of_list.
```

# A logician without ginkgo biloba

**Example (Killer Sudoku)**

The numbers may occur only once in each row, column and colored area if specified. In addition to Sudoku, a Killer Sudoku grid is divided into cages, shown with dashed lines. The sum of the numbers in a cage must equal the small number in its top-left corner. The same number cannot appear in a cage more than once.




1. Note the explicit negation required under the open world assumption
2. Domain size refers to the variables, not to the max value appearing in the formalisation

```
assign(domain_size,4).
assign(max_models,-1).
set(arithmetic).

formulas(latin_square).
  all x  all y1 all y2 (f(x,y1) = f(x,y2) -> y1 = y2).
  all x1 all x2 all y  (f(x1,y) = f(x2,y) -> x1 = x2).
end_of_list.

formulas(killer_sudoku).
  same_color(2,3) & same_color(0,1).  %let two sets {0,1} and {2,3}.
 -same_color(0,2).                    %0 and 2 are not from the same set
  all x same_color(x,x).                                    %reflexive
  all x all y (same_color(x,y) -> same_color(y,x)).         %symmetric
  all x all y all z (same_color(x,y) &                      %transitive
      same_color(y,z)  -> same_color(x,z)).

  all x1 all y1 all x2 all y2 (same_color(x1,x2) &    %Zone 2: up right
    same_color(y1,y2) & f(x1,y1) = f(x2,y2) &
    x1 + x2 >= 4 & y1 + y2 < 2 -> x1 = x2 & y1 = y2).

  all x1 all y1 all x2 all y2 (same_color(x1,x2) & %Zone 3: bottom left
    same_color(y1,y2) & f(x1,y1) = f(x2,y2) &
    x1 + x2 < 2 & y1 + y2 >= 4 -> x1 = x2 & y1 = y2).
end_of_list.

formulas(sample_puzzle_with_six_cages).
 f(0,2) + f(0,3)          = 3.                  %this cage has 2 cells
 f(0,0) + f(1,0)          = 5.                  %this cage has 2 cells
 f(0,1) + f(1,1) + f(1,2) = 3.                  %this cage has 3 cells
 f(1,3) + f(2,3) + f(2,2) = 3.                  %this cage has 3 cells
 f(2,0) + f(2,1) + f(3,0) = 4.                  %this cage has 3 cells
 f(3,1) + f(3,2) + f(3,3) = 6.                  %this cage has 3 cells
end_of_list.
```