CSP – CPP

MAI – UPC

José A. Magaña Mesa

January 2014

# Introduction

# Implementation

## Constraints

For a direct implementation of the Conway’s Game of Life rules, the following constraints must be added. On them, the matrix minimodel has been used so that manipulation of cells becomes straightforward.

As it was the case for the SAT problem, the board has been widened in order to avoid having to care about the position of the cell when looking for the neighbor cells.

For each Alive Cell in the original implementation, we assign the value:

rel(\*this, A(x,y), IRT\_EQ, 1); // the cell is alive

For all the cells in the original board (not extended), the neighborhood is calculated in the variable s, that adds the number of Alive Cells. The conditions for Alive Cells and Dead Cells are added. Operator >> implements the conditional in logic (🡪).

s=A(x-1,y-1)+A(x-1,y)+A(x-1,y+1)+A(x,y-1)+A(x,y+1)+A(x+1,y-1)+A(x+1,y)+A(x+1,y+1);

rel(\*this, (A(i,j)==0) >> (s!=3));

rel(\*this, (A(i,j)==1) >> (2<=s) && (s<=3));

Additional conditions are added on the side cells, to prevent propagation on a potentially infinite board:

// horizontal

rel(\*this, A(1,i)+A(1,i+1)+A(1,i+2)<3);

rel(\*this, A(n,i)+A(n,i+1)+A(n,i+2)<3);

// vertical

rel(\*this, A(i,1)+A(i+1,1)+A(i+2,1)<3);

rel(\*this, A(i,n)+A(i+1,n)+A(i+2,n)<3);

The margin of the extended board is set to zero (horizontally and vertically):

rel(\*this, A(i,0), IRT\_EQ, 0);

rel(\*this, A(i,n+1), IRT\_EQ, 0);

rel(\*this, A(0,i), IRT\_EQ, 0);

rel(\*this, A(n+1,i), IRT\_EQ, 0);

## Cost function

As the problem tries to maximize the number of Alive Cells, the cost function is directly the number of Alive Cells, that is, the sum of the values of the board. The minimum of the function is the number of Alive Cells in the initial configuration. The maximum can also be bounded based on the results of [1] and set the maximum to half the size of the board plus 1.

Cost=IntVar(\*this,nAlive,upperBound); // cost: minimum is the number of alive cells // maximum is the number of cells divided by 2

rel(\*this,Cost==sum(X));

## Variable and value selection

About variables:

INT\_VAR\_SIZE\_MAX

INT\_VAR\_ACTIVITY\_MAX

INT\_VAR\_ACTIVITY\_SIZE\_MAX

About values:

INT\_VAL\_MAX

INT\_VALUES\_MAX

INT\_VAL\_RANGE\_MAX

INT\_VAL\_SPLIT\_MAX

INT\_VAL\_MIN

INT\_VAL\_RND

# Evaluation

-print-last 1 -file-sol sl15\_2sol.txt -mode solution sl15\_2.txt

File-sol only one solution

File-stat only one stats

Settings incompatibility

Mode time considers iterations (others do not)

|  |  |  |  |
| --- | --- | --- | --- |
|  | INT\_VAR\_SIZE\_MAX | INT\_VAR\_ACTIVITY\_MAX | INT\_VAR\_ACTIVITY\_SIZE\_MAX |
| INT\_VAL\_MAX |  |  |  |
| INT\_VALUES\_MAX |  |  |  |
| INT\_VAL\_RANGE\_MAX |  |  |  |
| INT\_VAL\_SPLIT\_MAX |  |  |  |
| INT\_VAL\_MIN |  |  |  |
| INT\_VAL\_RND |  | ----- | ----- |

# Optimization

## Redundancies

# Conclusions

# References

Using relaxations in Maximum Density Still Life