INTRODUCTION TO ARTIFICIAL INTELLIGENCE

TASK 2: CONSTRAINT SATISFACTION PROBLEM

Misrraim Suárez Pérez - misrraimsp@gmail.com

1. CSP Formulation

(a) Letter Mosaic

- Variable. The fields of the board.
- Domain. The set of letters passed as an input for the program.
- Constrains. Two consecutive horizontal or vertical fields can not have the same value. More formally:

$$q_{ij} \neq q_{(i+1)j} \qquad q_{ij} \neq q_{i(j+1)}$$

(b) **Crossword Design**:

- Variable. There are *k* (input parameter that specifies the number of words to be allocated into the board) variables defined as a following set of information:

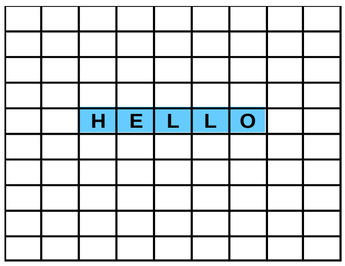
{int **startRow,** int **startColumn,** byte **direction,** String **word}**

- Domain. The domain for this problem is built as the all possible combinations for the particular data fields of the variable. These particular data fields subdomains are:

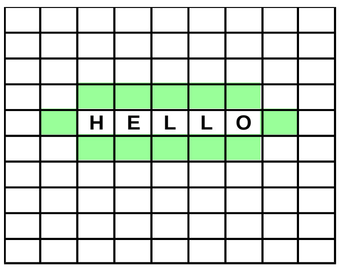
{[1 - rowNum], [1 - colNum], [0 - 1], {S}}

- Constrains. For the specification of the constraints three sets are defined.

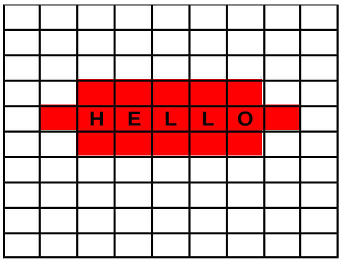
(1) Core. The set of fields that the word occupy on the board



(2) Border. The set of fields that surround the word in the board



(3) Influence. The union of Core and Border sets



The constraints are defined as binary relations between all of the *k* variables, in such a way that:

- The word field must be different and
- If: words have the same direction (they are parallel) Then: the intersection between Influence set and Core set must be empty.
- Else: or Core sets intersection has one element, or Border sets intersection is empty

2. Parameter Testing

In this section is presented a brief discussion on performance of both algorithm, backtrack and forward checking, by passing them different parameters. The performance is measured as the time needed, in ms, until reaching the first solution, or after reaching the no solution state by going throughout the hole tree.

There are two tables, one for BT data and another for FC ones. In both tables appear the value Inf, that need to be interpreted as infinite in the sense that the execution time was huge and it was necessary to abort the process.

In columns it is possible to see the evolution of performance when the number *K* of words to be allocated into the table is variable, for a fixed board size.

In rows it is possible to appreciate the evolution of performance when the number K is fixed and it is clanged the size of the board.

Back Tracking

BT	4 (2x2)	16 (4x4)	64 (8x8)	256 (16x16)
2	7	3	0	32
4	90	26	31	62
8	91	inf	110	782
16	79	inf	inf	1140

Forward Checking

FC	4 (2x2)	16 (4x4)	64 (8x8)	256 (16x16)
2	1	68	719	4859
4	34	119	2672	32896
8	70	inf	7844	123445
16	69	inf	inf	145888

The most relevant conclusion that can be taken is that Forward-checking performance does not improve simple Backtrack. The simplest BT algorithm is faster.

This can be explained if we see at the huge domains (very easy to have billion-size domains or greater; domains explode with board size and set S of words) that this problem has. In forward-chacking it is necessary to check all the values of the domain of all non-assigned variables, which is determinant in this problem.

3. References

GraphStream Library: http://graphstream-project.org

Russell and Norvig: http://aima.cs.berkeley.edu/2nd-ed/newchap05.pdf