CSC 403, Assignment 3

Due on October 31st at 11:59 pm

In this assignment you will create small crossword puzzles starting from a given set of words. Here is the problem statement:

> A word is defined as a list of letters. The set of words (or dictionary) is given by a collection of PROLOG facts defining the predicate `word/1`. One such a dictionary is given in the file cross.pl. Six words from the dictionary must be arranged in a 3×3 crossword puzzle (three across and three down) while the following constraint must be observed: for any two words, the letter where they intersect must be the same.

You must approach the problem from two perspectives:

1. Assign one PROLOG variable for each square in the grid, such as `Xij` for grid square (i, j). Then, row j will be `[X1j, X2j, X3j]` while column j will be `[Xj1, Xj2, Xj3]`. Then match the dictionary with this kind of lists, thus solving the problem.
2. Formulate and then use PROLOG to solve the problem at hand as a state space search problem. In addition to the program itself provide (in a comment) the query that will need to be issued to obtain the solution.

How many crossword puzzles are produced? Depending on the dictionary that is provided as input, the number n of answers may be different. What is the property that holds for n no matter what is the input? Experiment, find out, and explain.

Hints

You may notice that SWI Prolog prints only a truncated version of the result whenever the whole result it is too long. To see the whole result use the predicate `writelist/1` provided in the file cross.pl. In other words, instead of

```
?- search(..., R).
```

use the following query:

```
?- search(..., R), writelist(R).
```

The following hints apply to Question 2:

There are six locations in the puzzle (across-1, across-2, across-3, down-1, down-2, and down-3), all of them having the whole set of words as domains. A location can be assigned a word or

not. You can therefore represent a state as a list of assigned location (and their assigned words) plus a list of locations not yet assigned[1]. The latter can be represented explicitly or implicitly.

In order to obtain the letter *J* of some word *W*, you can use the predefined predicate `nth1` (the last character is the number 1, not the letter l!). Given some number `N` and some list `L`, the query `nth1(N,L,R)` binds `R` to the `N`-th element of `L`, considered numbered from 1; the query fails if the length of `L` is smaller than `N`. Examples are given in the file cross.pl.

Submission guidelines

Provide your scripts and the plain English answers to questions as appropriate. Also provide sessions listings that test your functions (*any untested part of your submission will be disregarded*).

Please do not submit anything else than plain text. Ideally, everything should be submitted in one text file containing your PROLOG code and including the rest of the required deliverables as comments. Note that in PROLOG the character `%` is used as comment prefix, but C-style block comments (`/*...*/`) are also recognized. Provide your submission by email.

Recall that assignments can be solved in groups[2] (of maximum three students), and a single solution per group should be submitted. Also recall that a penalty of 10% per day will be applied to late submissions.

Ensure that you name your submission files appropriately with the first four letters of your last name, then the first three letters of your first name, and finally "Assign_0X" at the end. For example, Mier_Gre_Assign_01.pl.