Artificial Intelligence (AI)

Lab Sheet No: 2

Introduction

Constraint programming is a useful tool in formulating and solving problems that can be defined in terms of constraint among a set of variables. In fact real world problems are defined in terms of some variables that bear some constraints. Finding a set of variables that are within the constraints given (or observed) is a solution to that problem.

Let us consider a problem, that can be represented by some relations of the variables x, y and z. We have a domain Dx, Dy, Dz from where the variables can take a value. The constraint is given by a set C and may have a number of constraints C1, C2, C3 etc each relating some or all of the variables x, y, z. Now a solution (or solutions) to the problem is a set of the problem is a set dx∈Dx, dy∈Dy, dz∈Dz and all the constraints of set C are satisfied.

Crypto arithmetic problem

Crypto Arithmetic Problem is yet another constraint satisfaction problem. We have to assign numeric values (0 through 9) to the alphabets in the given words in such a way that the sum of the two words equals the third.
For example, **SEND+MORE=MONEY**
We have to assign values to the individual alphabets in such a way the arithmetic rules are followed, a trivial solution will be assign zeros to all but we have a constraint, no two alphabets should be assigned with the same number.

C4	C3	C2	C1	
	S	E	N	D
+	M	O	R	E
M	O	N	E	Y

Now domain for alphabet is given by:
S, E, N, D, M, O, R, Y∈{0,1,2,3,4,5,6,7,8,9}.
The constraints are:
D+E=Y+10C1
N+R+C1=E+10C2

E+O+C2=N+10C3
S+M+C3=O+10C4
M=C4
C1, C2, C3, C4∈{0,1}
And we have the constraint that no two alphabets should be assigned to the same number.

PROGRAM: 1

```
DOMAINS
int_list=integer*

PREDICATES
solution(int_list)
member(integer,int_list)

CLAUSES
solution([]).
solution([S,E,N,D,M,O,R,Y]):-
     C4=1, %c4 must be 1 otherwise no need to mention M.
     member(C1,[0,1]),
     member(C2,[0,1]),
     member(C3,[0,1]),
     % C1, C2, C3 will have values 0 or 1
     member(E,[0,1,2,3,4,5,6,7,8,9]),
     member(N,[0,1,2,3,4,5,6,7,8,9]),
     member(D,[0,1,2,3,4,5,6,7,8,9]),
     member(M,[0,1,2,3,4,5,6,7,8,9]),
     member(O,[0,1,2,3,4,5,6,7,8,9]),
     member(R,[0,1,2,3,4,5,6,7,8,9]),
     member(Y,[0,1,2,3,4,5,6,7,8,9]),
     member(S,[0,1,2,3,4,5,6,7,8,9]),
     % S,E,N, D, M, O, R, Y will have values between 0 and 9.
     % The values of S,E,N, D, M, O, R, Y  must not be equal.

     S<>E, S<>N, S<>D, S<>M, S<>O, S<>R, S<>Y,
     E<>N, E<>D, E<>M, E<>O, E<>R, E<>Y,
     N<>D, N<>M, N<>O, N<>R, N<>Y,
     D<>M, D<>O, D<>R, D<>Y,
     M<>O, M<>R, M<>Y,
     O<>R, O<>Y,
```

```
R<>Y,
% Computation for solution
D+E=Y+10*C1,
N+R+C1=E+10*C2,
E+O+C2=N+10*C3,
S+M+C3=O+10*C4,
M=C4.

member(X, [X|_]).
member(X, [_|Z]):-
     member(X,Z).

GOAL
solution([S,E,N,D,M,O,R,Y]).
```

Assignment (1): Make yourself a crypto arithmetic problem as above and find the solution.

Eight Queens Problem

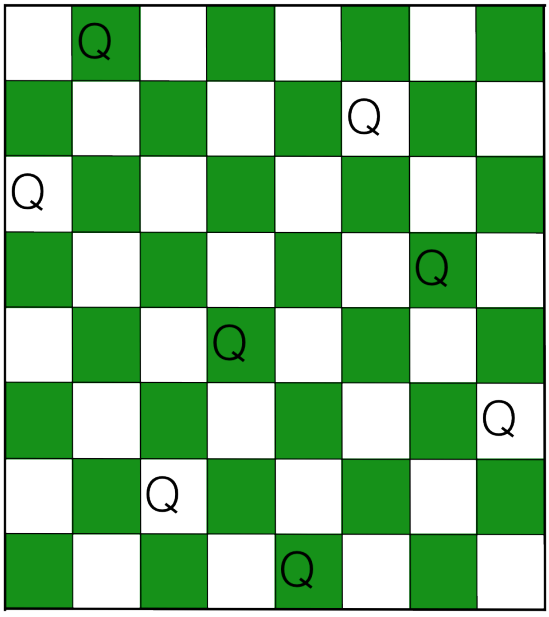
Eight queens problem is a constraint satisfaction problem. The task is to place eight queens in the 64 available squares in such a way that no queen attacks each other. So the problem can be formulated with variables $x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8$ and y_1, $y_2,y_3,y_4,y_5,y_6,y_7,y_8$; the xs represent the rows and ys the column. Now a solution for this problem is to assign values for x and y such that the constraint is satisfied.

The problem can be formulated as P = $\{(x_1, y_1), (x_2, y_2)$ ……$(x_8, y_8)\}$ where (x_1, y_1) gives the position of the first queen and so on.

So it can be clearly seen that the domains for x_i and y_i are Dx = {1,2,3,4,5,6,7,8} and Dy = {1,2,3,4,5,6,7,8} respectively.

The constraints are

1) No two queens should be in the same row. That is, $y_i \neq y_j$ for i = 1 to 8; j = 1 to 8; $i \neq j$
2) No two queens should be in the same columns.That is, $x_i \neq x_j$ for i = 1 to 8; j = 1 to 8; $i \neq j$



3) There should not be two queens placed on the same diagonal line.
   That is, $(y_i - y_j) \neq \pm (x_i - x_j)$.

Now a solution to this problem is an instance of P wherein the above mentioned constraints are satisfied.

PROGRAM: 2

```
DOMAINS
cell = c(integer, integer)
list = cell*
int_list = integer*
PREDICATES
solution(list)
member(integer, int_list)
noattack(cell,list)

CLAUSES
solution([]).

solution([c(X,Y)|Others]):-
      solution(Others),
      member(Y,[1,2,3,4,5,6,7,8]),
      noattack(c(X,Y),Others).

noattack(_,[]).
noattack(c(X,Y),[c(X1,Y1)|Others]):-
      Y<>Y1,
      Y1-Y<>X1-X,
      Y1-Y<>X-X1,
      noattack(c(X,Y),Others).
member(X,[X|_]).
member(X,[_|Z]):-
      member(X,Z).

GOAL
solution([c(1,A),c(2,B),c(3,C),c(4,D),c(5,E),c(6,F),c(7,G),c(
8,H)]).
```

Assignment (2):

Observe the result of the above program and discuss on the result. Test the goal by placing a few queens explicitly.
(Try goals like solution ([c(1,1),c(2,B),c(3,C),c(4,8),c(5,E),c(6,F),c(7,G),c(8,H)] etc.)

Assignment (3):

Try to solve the above problem using C, C++, .NET or Java.

Assignment 4): Create a DLL file using prolog and use it in any of the available languages java or .NET to depict the solution. Make it interactive. (You may want to set a queen in the fifth row of the first column)