TRIBHUVAN UNIVERSITY
INSTITUTE OF ENGINEERING
PULCHOWK CAMPUS

REPORT
ON

Submitted By:

Name: Kripesh Nihure
Class: 077BCT AB
Roll: 037

Submitted To:

Department of Computer Engineering, Pulchowk Campus

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.

1. 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.

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.

For example, SEND+MORE=MONEY

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.

Assignment 1

Make yourself a crypto arithmetic problem as above and find the solution.

Let the problem be as below:

```
   C6 C5 C4 C3 C2 C1
   B  A  N  A  N  A
+  G  U  A  V  A
-------------------
O  R  A  N  G  E
```

```
DOMAINS
int_list=integer*

PREDICATES
solution(int_list)
member(integer, int_list)

CLAUSES
solution([]).
solution([B,A,N,G,U,V,O,R,E]):-
        member(C1, [0]),
        member(C2, [0,1]),
        member(C3, [0,1]),
        member(C4, [0,1]),
        member(C5, [0,1]),
        member(C6, [0,1]),
        member(B,[0,1,2,3,4,5,6,7,8,9]),
        member(A,[0,1,2,3,4,5,6,7,8,9]),
        member(N,[0,1,2,3,4,5,6,7,8,9]),
        member(G,[0,1,2,3,4,5,6,7,8,9]),
        member(U,[0,1,2,3,4,5,6,7,8,9]),
        member(V,[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(E,[0,1,2,3,4,5,6,7,8,9]),
```

```
B<>A, B<>N, B<>G, B<>U, B<>V, B<>O, B<>R, B<>E,
A<>N, A<>G, A<>U, A<>V, A<>O, A<>R, A<>E,
N<>G, N<>U, N<>V, N<>O, N<>R, N<>E,
G<>U, G<>V, G<>O, G<>R, G<>E,
U<>V, U<>O, U<>R, U<>E,
V<>O, V<>R, V<>E,
O<>R, O<>E,
R<>E,

C1+2*A=E+10*C2,
C2+N+V=G+10*C3,
C3+2*A=N+10*C4,
C4+N+U=A+10*C5,
C5+A+G=R+10*C6,
C6+B=O.
member(X, [X|_]).
member(X, [_|Z]):-
member(X,Z).
```

GOAL

```
solution([B,A,N,G,U,V,O,R,E]).
```

Output :

B=8, A=7, N=5, G=6, U=2, V=1, O=9, R=3, E=4

Discussion

The Prolog program efficiently encodes the constraints of the cryptarithmetic problem, employing logical predicates and arithmetic relations. The use of backtracking in Prolog enables systematic exploration of possible digit assignments, while the member predicates restrict variable values to the valid digit set.

2. Eight Queen’s 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 x1,x2,x3,x4,x5,x6,x7, and y1,y2,y3,y4,y5,y6,y7,y8; 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 = {(x1, y1), (x2, y2).....(x8, y8)}

where (x1, y1) gives the position of the first queen and so on.

So it can be clearly seen that the domains for xi and yi 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.

Thatis, yi ≠ yj for i = 1 to 8; j = 1 to 8; i ≠ j

2) No two queens should be in the samecolumns.

That is, xi ≠ xj for i = 1 to 8; j = 1 to 8; i ≠j

3) There should not be two queens placed on the same diagonal line.

That is, (yi – yj) ≠ ± (xi –xj).

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

For 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)]

OUTPUT

B = 7 , C = 5 , E = 2 , F = 4 , G = 6 , H = 3

Discussion

The Prolog program efficiently solves the Eight Queens Puzzle by employing logical constraints and backtracking. It utilizes the ‘solution‘ predicate to place queens on the chessboard, ensuring no two queens threaten each other. The ‘noattack‘ predicate enforces constraints on the queens’ positions, preventing conflicts in rows, columns, and diagonals. The output provides a specific placement solution, such as A=1, B=5, C=8, D=6, E=3, F=7, G=2, H=4, indicating the column positions for each queen. The program showcases Prolog’s suitability for expressing and solving problems through declarative logic.

Assignment 3&4

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

C++ CODE

```
#include <bits/stdc++.h>
using namespace std;

int board[8][8];
bool isPossible(int n, int row, int col)
{
for (int i = row - 1; i >= 0; i--)
        if (board[i][col] == 1)
                return false;

for (int i = row - 1, j = col - 1; i >= 0 && j >= 0; i--, j--)
{
if (board[i][j] == 1)
        return false;
}
for (int i = row - 1, j = col + 1; i >= 0 && j < n; i--, j++)
{
if (board[i][j] == 1)
        return false;
}

return true;
}
void nQueenHelper(int n, int row)
{
if (row == n)
{
for (int i = 0; i < n; i++)
{
for (int j = 0; j < n; j++)
        cout << board[i][j] << " ";
}
cout << endl;
return;
}

for (int j = 0; j < n; j++)
{
```

```
if (isPossible(n, row, j))
{
board[row][j] = 1;
nQueenHelper(n, row + 1);
}
board[row][j] = 0;
}

return;
}
void placeNQueens(int n)
{
memset(board, 0, 8 * 8 * sizeof(int));
nQueenHelper(n, 0);
}

int main()
{
n=8;
placeNQueens(n);
return 0;
}
```


Output

```
1 0 0 0 0 0 0 0
0 0 0 0 0 0 1 0
0 0 0 0 1 0 0 0
0 0 0 0 0 0 0 1
0 1 0 0 0 0 0 0
0 0 0 1 0 0 0 0
0 0 0 0 0 1 0 0
0 0 1 0 0 0 0 0
```

Discussion

To solve the eight queen's problem, we used the Backtracking algorithm. The backtracking algorithm, in general checks all possible configurations and test whether the required result is obtained or not. For the given problem, we will explore all possible positions the queens can be relatively placed at. The solution will be correct when the number of placed queens = 8.

Conclusion

Hence, we successfully solved cryptarithmetic puzzles and the N-Queens problem using Prolog's logical programming for constraint satisfaction and Python, Java, and C++ for versatile and efficient algorithmic solutions.