CMPUT 325 Wi17 - NON-PROCEDURAL PROG LANGUAGES Combined LAB LEC Wi17

Assignment 4 Winter 2017

CMPUT 325 - Assignment 4

Due Date: Thursday, April 13, 11:55pm. Submit as a single file <ID>.pl .

There are two questions in this assignment, each worth 10 assignment marks. For each of the questions, make sure your program doesn't get into an infinite loop. You are free to use either pure Prolog, or CLP(FD) for this assignments. There are no restrictions, so you can use the cut and negation operators.

Question 1 - Four squares

It's an interesting theorem from number theory that every natural number can be expressed as the sum of four or fewer squares. For example:

```
1 = 1*1
2 = 1*1 + 1*1
3 = 1*1 + 1*1 + 1*1
4 = 2*2
5 = 1*1 + 2*2
6 = 2*2 + 1*1 + 1*1
7 = 2*2 + 1*1 + 1*1 + 1*1
8 = 2*2 + 2*2
```

Implement the predicate

```
fourSquares(+N, [-S1, -S2, -S3, -S4])
```

*Note that, as a convention, when we specify a predicate, a parameter prefixed with + means it is an input parameter--it's bound when the predicate is called, and a parameter with - is an output parameter--it's a variable when the predicate is called and bound to a value when the predicate is proved.

Given any positive integer N greater than 0, it returns a list of non-negative integers [S1,S2,S3,S4] such that

```
N = S1*S1 + S2*S2 + S3*S3 + S4*S4.
```

In order to keep things nice and tidy, it is also required that S1 <= S2 <= S3 <= S4. Note that some of S1...S3 may be zero.

On backtracking, fourSquares should return all possible answers since there may be more than one solution, e.g.

```
20 = 2*2 + 4*4 and
20 = 1*1 + 1*1 + 3*3 + 3*3
```

The order in which multiple solutions are returned is up to you. We will use findall to collect all solutions from your predicate.

Hint: For pure Prolog solutions, it may be useful to define a predicate count(+N,+M,-X) which given two integers N and M returns all the integers between them in succession. For example, the query

```
?- count(3,7,X)
```

would return

```
X = 3;
X = 4;
X = 5;
X = 6;
X = 7;
```

Question 2 - War and Peace

Two countries have signed a peace treaty and want to disarm over a period of months, but they still don't completely trust each other. Each month one of the countries can choose to dismantle one military division while the other can dismantle two. Each division has a certain strength, and both sides want to make sure that the total military strength remains equal at each point during the disarmament process.

For example, suppose the strenghs of the country's divisions are:

```
Country A: 1, 3, 3, 4, 6, 10, 12
Country B: 3, 4, 7, 9, 16
```

One solution is:

```
Month 1: A dismantles 1 and 3, B dismantles 4
Month 2: A dismantles 3 and 4, B dismantles 7
Month 3: A dismantles 12, B dismantles 3 and 9
Month 4: A dismantles 6 and 10, B dismantles 16
```

Write a predicate

```
disarm(+Adivisions, +Bdivisions,-Solution)
```

Where Adivisions and Bdivisions are lists containing the strength of each country's divisions, and Solution is a list describing the successive dismantlements. In the example above,

```
Solution = [[[1,3],[4]], [[3,4],[7]], [[12],[3,9]], [[6,10],[16]]]
```

Each element of Solution represents one dismantlement, where a dismantlement is a list containing two elements: the first element is a list of country A's dismantlements and the second is a list of country B's dismantlements.

Finally, the countries want to start with small dismantlements first in order to build confidence, so make sure that the total strength of one month's dismantlement is less than or equal to the total strength of next month's dismantlement.

We will only evaluate the first solution computed by your program. If there is no solution, then your program should return false.

Submission status

This assignment will accept submissions from **Friday, 7 April 2017, 12:00 AM**

Attempt number

This is attempt 1.

Submission status

No attempt

Grading status

Not graded

Due date

Thursday, 13 April 2017, 11:55 PM

Time remaining

16 days

Last modified

Tuesday, 28 March 2017, 11:18 PM

Submission comments

▸ Comments (0)