Advanced Programming

Assignment 3: Analyzing TwitBook

Per Steffen Czolbe, Konrad Gnoinski

Handed in: October 4, 2017

Contents

Description of the Assignment

```
import numpy as np

def incmatrix(genl1,genl2):
    m = len(genl1)
    n = len(genl2)
    M = None #to become the incidence matrix
    VT = np.zeros((n*m,1), int)  #dummy variable

    #compute the bitwise xor matrix
    M1 = bitxormatrix(genl1)
    M2 = np.triu(bitxormatrix(genl2),1)

    for i in range(m-1):
        for j in range(i+1, m):
            [r,c] = np.where(M2 == M1[i,j])
            for k in range(len(r)):
                VT[(i)*n + r[k]] = 1;
                VT[(i)*n + c[k]] = 1;
                VT[(j)*n + r[k]] = 1;
                VT[(j)*n + c[k]] = 1;
\\\\
                if M is None:
```

```
            M = np.copy(VT)
        else:
            M = np.concatenate((M, VT), 1)

        VT = np.zeros((n*m,1), int)

    return M
```

This assignment is about Analyzing TwitBook which is a a set of relationships between group of people.

Solution

This section gives a brief overview about the structure of the solution and how to execute the code.

Implementation

This solution of the assignment is implemented in a functional programming language - Prolog. SWIProlog is used in this assignment.

Source-code Organisation

The code is organised in just two files:

- `src/twitbook.pl`: the file containing all the required predicates as well as definition of twitbook structures.
- `src/tests.pl`: the file containing static tests of the predicates functionality.

Test Cases

The `src/test.pl` file contains multiple tests. The test framework used is `PL-Unit`. In order to validate the tests :

```
$ [twitbook], [tests].
$ run_tests.
```

How to Execute the Program

The predicates can be imported from the `src/twitbook.pl` file. Our predicates were made with use of SWIProlog. Calling predicates is performed by e.g. :

```
$ [test], [twitbook].
$ g1(G), likes(G, kara, oliver).
```

Implementation

A description of the implementation.

Assumptions

We assume that graph always would be given and our code would not be asked to generate graph with use of other parameters of the predicate. Mainly, it does not make a common sense to generate graph with use of e.g. likes(G,X,Y) predicate, when only X and Y would be provided.

Assessment

This section contains the assessment of the presented solution.

Correctness of Solution

Based on the extensive tests, analyzing the code we assume the solution is likely to be correct.
Additionally, the solutions passed the Online TA test.

Summary of Code Quality

Based on the extensive tests, we do believe that the functionality of this set of predicates matches the functionality specified in the assignment.

The quality of the code could be improved by using more functions from the Prolog standard library, especially using some of the restricted operators. However, that is our first experience in the new language, so we find it beneficial to be restricted from use of build in predicates. That we need to fully understand basics of a language, without being artificially pushed further by the use of complex predicates that are a closed box for us. Predicate `different_world` was not implemented.