CS4093 Special Topics in Computing: Constraint Programming and Optimisation

1st Continuous Assessment 2015/16

A shipment of industrial waste products has arrived at Cork harbour, and must be taken by secure trucks to a processing plant to be safely neutralised. We have a limited number of trucks available to us. Each hazardous item comes in an individual box, and all boxes are the same size. Each truck can carry up to the same number of items. There are some safety regulations on these truck loads: if there is an accident, some items would react with each other, and cause an explosion or a hazardous gas leak, and so some pairs of items are forbidden from being carried at the same time in the same truck. Our task is to assign all of the items to the trucks so that all items are loaded, no truck is overloaded, and no truck contains a forbidden pair of items.

Problem instances are described in a standard format in text files:

```
n t c m
x1 y1
x2 y2
...
xm ym
```

where *n* is the number of items, *t* is the number of trucks, *c* is the capacity of the trucks (i.e. maximum number of items able to be loaded), and *m* is the number of forbidden pairs. There are then *m* rows each containing two integers, where each row xi yi indicates that item xi and item yi cannot be loaded onto the same truck.

For example, in the file "multitrucks1.txt":

```
9 2 5 5
0 4
1 3
4 5
6 7
6 8
```

there are 9 items, 2 trucks, each with capacity 5, and there are 5 forbidden pairs. Item 0 cannot be loaded with item 4, item 1 cannot be loaded with item 3, etc..

A class MultiTruckDataLoader.java is provided, which will read the problem instance from a data file, and supports the following public getter methods:

```
public int getnItems();
public int getnTrucks();
public int getCapacity();
public int[] getnPairs();
public int[][] getbadpairs();
```

The assignment

1. Create a Choco class for modeling and solving problems as described above. Your class must be called "MultiTrucks.java", it must have an executable main method which should throw an IOException), and it must read in data from text files in the above format. Include in your submission any other Java classes that you wrote that your main class needs to compile and

run. If you do not use multiTruckDataLoader.java, then you must include your own method for reading the files.

2. Write a brief report describing your models, explaining any different ways of representing the problem that you considered and why you chose the one you did, and showing the results you get when you run you Java/Choco code on the three data files. The report should be formatted as a plain text file (.txt).

Submission

All files and reports must be contained in a single zipped tar file, and emailed to Ken Brown (k.brown@cs.ucc.ie) by the submission deadline.

Marking scheme

Marks will be awarded for submitting a working Java/Choco program, for generating correct solutions, for the quality of model, for good use of Choco facilities including global constraints if needed, and for the explanation in the report. Marks will not be awarded for the quality of the Java code that supports the model, so don't waste time implementing a GUI to show the results (although if your code is inadequately commented and is difficult to understand, you may lose marks).

Submission Deadline

5pm, Friday 5th February, 2016.

Plagiarism

This submission is part of the formal assessment for the module, and so must be your own work. You should not submit anyone else's work as if it were your own, and you should not be collaborating with each other on how to solve the problem. However, you are free to re-use, without acknowledgement, any piece of Java/Choco code that has been posted on the CS4093 web pages or included in CS4093 Lecture slides, or included in the Choco manual.