Department of Mathematics and Computer Science
University of Southern Denmark, Odense

September 10, 2013

Marco Chiarandini

DM811 - Heuristics for Combinatorial Optimization

Assignment 1, Autumn 2013

Submission deadline: Wednesday, September 18, 2013 at 12:00.

Read all this document before starting to work.

- The task of the assignment is to develop a randomized construction heuristic for solving the chromatic number problem defined in Assignment 0.
- The heuristic must be a single pass heuristic without backtracking and it must contain some element of randomness so that if rerun on the same instance chances are that the solution will be different.

  Each run must be reproducible. Thus the program must take a random seed as an input from command line and initialize the random generator with it.
- The format of input files and solutions remains the same as for Assignment 0. As test instances you can use the ones from Assignment 0. However, the programs will be evaluated on new unseen instances: the N^2-queens graphs, with N from 4 to 32.

  On all instances your program will be halted after one minute of running time, but it should finish much earlier.
- You can submit your program more than once in which case, the last submission will count. You will compete with your colleagues on both running time and solution quality.
- You must also submit a written description of your construction heuristic of max 3 A4 pages. Do not put your name on the document but rather use a pseudonym. The descriptions will be peer reviewed.

  If you implemented and tested more than one heuristic, describe only the one that you submitted and say how you have chosen it.
- In order to pass the assignment, your program must return valid results to all test instances and hence appear in the page of the analysis of results.

Submission instructions

The part below is copied from Assignment 0, nothing has changed. The description of the algorithm must be placed in `doc/`.
Start early to test your submission. You can submit as many times as you wish within the deadline. Every new valid submission overwrites the previous submission.
Your archive must decompress as follows:

- `README`
- `bin/` where your executable called `gcp` must be
- `src/` the source files that implement the algorithm
- `Makefile` to compile the sources in `src` and puts the executables in `bin`.

Your program will NOT be recompiled; the executable file `gcp` in `bin/` will be used for the tests. `Makefile` and `src/` are required just for debugging purposes in special cases. Additionally, you are recommended to have the following content, which will however not be used.

- `data/` containing the test instances.
- `res/` containing your results, the performance measurements
- `r/` statistics, data analysis, visualization
- `doc/` a description of the algorithms.
- `log/` other log files produced by the run of the algorithm

The programs will run on a machine with ubuntu 12.04, hence you can log in on any machine of the terminal room, compile your program there, and make sure that it executes.
If your program is written in C or C++ you should compile with the `-lm` and `-static` flags.
If your program is written in Java then your bin directory must contain a jar file that you compiled on an IMADA machine and a shell executable file called `gcp` containing:

```
#!/bin/bash
HERE=‘dirname $0‘
java -jar $HERE/gcp.jar $@
```

Python users can do the same but with python commands instead.

The executable must run with the following arguments:

- `-i filename` the instance file
- `-c filename` the file with the solution in the format described below
- `-s #` a random seed
- `-t #` an integer indicating the time limit in seconds.

for example:

```
gcp -i ../data/marco10.col -c marco10.sol -s 10 -t 60
```

All output must go in the standard output and in the solution file. Do not create other files. The files containing the solution must be named with the name of the instance (without `.col` extension) and with extension `.sol`. The files must be in ASCII format and consist of a single column of numbers representing the colors to each vertex. Each number indicates the color for the vertex corresponding to the line number. Both colors and vertices start at 1. For example:

```
$ cat marco10.sol
2
3
4
3
2
2
1
...
```

indicates that the vertex 1 receives color 2, vertex 2 receives color 3, etc.
Note that the upload and analysis system is still under test and therefore it may have to be fine tuned, hence be patient and contact the teacher if something is not working as it should.