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

December 14, 2008
Marco Chiarandini

DM812 - Metaheuristics

Exam Project, Second Quarter, Fall 2008

Note 1 The project is carried out individually and it is not allowed to collaborate. It consists of algorithm design, implementation, experimentation and written report.

The evaluation of the project is based on the report only. A program that implements the algorithms described in the report must also be submitted. However, the program will be checked only for verifying the correctness of the results presented. The report may be written in either English or Danish.

Note 2 Corrections or updates to the project description will be published on the course web page and will be announced by email to the addresses available in the Blackboard system. In any case, it remains students' responsibility to check for updates on the web page.

Note 3 *Submission*. An archive containing the electronic version of the written report and the source code of the program must be handed in through the Blackboard system **before 12:00 of Monday, January 19, 2009**. This is the procedure:

- choose the course DM812 in Blackboard,
- choose "Exam Project Submission" in the menu on the left,
- fill the form and conclude with submit,
- print the receipt (there will be a receipt also per email).

See Appendix for details on how to organize the electronic archive. Reports and codes handed in after the deadline will generally not be accepted. System failures, illness, etc. will not automatically give extra time.

---

1 Project Description

The theme of this project is applying metaheuristics for solving the Maximum Stable Set Problem on Hypergraphs. The project builds on the project of DM811, Fall 2008 [Chio8]. The text of that project constitutes an integrand part of the current document, hence refer to [Chio8] for a statement of the problem and a description of the test instances that have to be solved. Recall that there are four classes of instances obtained by the combination of the size of edges (10 or 50) and the number of edges (1000 or 10000). Refer to the web page of DM811 for downloading these instances and a program to verify solutions. Input and output formats remain those described in [Chio8].

The project consists of five *tasks* each one devoted to the application of one of the metaheuristics studied in the course to the Maximum Stable Set Problem on Hypergraphs. The metaheuristics relative to the five tasks are:

1. Simulated annealing
2. Scatter search **or** path relinking
3. Evolutionary algorithm
4. Estimation of distribution algorithm **or** cross entropy
5. Ant colony optimization

Each of the five tasks above can be further divided into three *subtasks*:

- design, possibly including alternative choices for the metaheuristic components.
- implementation
- experimental analysis, using the methods treated in the lecture (for example, the F-race).

In order to consider covered one of the tasks above, each of the three subtasks must be accomplished and described in the report. The three subtasks are accomplished when a version of a metaheuristic has been implemented, tuned on the instances available and the null-metaheuristic outperformed [BD06]. Failure to fulfill one (or more) of the five tasks may lead to an insufficient performance. If some task is not addressed, then there must be some in-depth study on the other tasks. This includes comparing, in a sound way, different reasonable choices for the components.

In the experiments, the computation time of a single run of a metaheuristic must not exceed 300 seconds. However, if this leads to time issues in running the experiments, shorter runs, say 120 seconds as time limit, can also be acceptable, provided this time is enough to observe an effect of the metaheuristic.[1]

2 Remarks

Remark 1 The subtask "design" is intrinsic to both implementation and experimental analysis. Preference will be given to a component-wise approach that allows to identify the contribution of each components. A component-wise organization and presentation is a good choice also in the written report.

Remark 2 For the metaheuristics at points 3, 4 and 5 the use of local search subroutines is not allowed.

Remark 3 In all of the five tasks, the primary goal is to show that the principles underlying the metaheuristics have an effect on the quality of the solutions found. Therefore, it is considered a good design choice removing any confounding factor in the algorithm such that the contribution of the main components of the metaheuristic arise. For example, the use of procedures to make feasible a solution generated by sampling a probability distribution is acceptable, but an even better choice is designing the algorithm in such a way that the solution produced by the sampling procedure is already feasible.

[1] Computation times refer to an Intel Core 2 CPU 6300 at 1.86GHz, with 2048 KB cache and 2 GB RAM, running an Ubuntu 8.04 distribution of Linux with kernel 2.6.24-19-generic.

Remark 4 Contrary to DM811, the quality of results achieved will not necessarily have a strong influence on the final grade. Other aspects will be deemed more important in the evaluation. In particular, it is important to show that the choices made for the various components of a metaheuristic are reasonable and outperform experimentally other naive choices (for example, the null-metaheuristic). Methods, comments and conclusions in the analysis of the results of experiments will also contribute to the final evaluation more than the quality of the results.

Remark 5 The total length of the report should not be less than 10 pages and not be more than 18 pages, appendix included (lengths apply to font size of 11pt and 3cm margins). Although these bounds are not strict, their violation is highly discouraged.

Remark 6 In the description of the algorithms (or parts thereof), it is allowed to use short algorithmic sketches in form of pseudo-code, decision trees, flow charts and general local search machines but it is not allowed to include source codes.

Remark 7 This is a list of factors that could lead to a better grade:

- sound criticism in the comments of results;
- originality of the approaches chosen;
- originality of the experimental questions;
- an organization and description of experiments that guarantee reproducibility;
- writing style;
- effective use of graphics in the presentation of experimental results.

Electronic Submission

The electronic submission must be organized similarly to what done for DM811, but all files must be maintained *anonymous*. This means, in particular, that no name or CPR-number must be written on the document containing the report. The organization of the archive is described in the following. The archive must expand in a main directory:

```
DM812/
```

and its content:

```
DM812/README
DM812/Report/
DM812/src/
```

The file README contains the manual for the compilation of the program. The directory src contains the sources which may be in C, C++, Java or other languages. If needed a Makefile can be included either in the root directory or in src. After compilation the executable must be placed in `src`. For java programs, a jar package can also be submitted.

Programs must work on IMADA's computers under Linux environment and with the compilers and other applications present on IMADA's computers. Students are free to

develop their program at home, but it is their own responsibility to transfer the program to IMADA's system and make the necessary adjustments such that it works at IMADA.[2]

The executable must be called mss. It will be run by typing in the directory `DM812/src/`:

```
mss -i INSTANCE -t TIME -s SEED -o OUTPUT
```

- `-i INSTANCE` to load the data associated with the file INSTANCE.
- `-m NUMBER` which metaheuristic. The number refer to the task number (see description above).
- `-t TIME` to stop the program execution after TIME seconds.
- `-s SEED` seed to initialize the random generator.
- `-o OUTPUT` the file name where the solution is written

For example:

```
mss -i u-100-50-1000-01.mss -m 1 -o u-100-50-1000-01.sol -t 180 -s 1
```

will run simulated annealing on the instance `u-100-50-1000-01.mss` opportunely retrieved from the given path for 180 seconds with random seed 1 and write the solution in the file `u-100-50-1000-01.sol`.

It is advisable to have a log of algorithm activities during the run. This can be achieved by printing further information on the standard error or in a file. A suggested format is to output a line whenever a new best solution is found containing at least the following pieces of information:

```
best 853 time 10.000000 iter 1000
```

All process times are the sum of user and system CPU time spent during the execution of a program as returned by the linux library routine `getrusage`. Process times include the time to read the instance.

References

[BD06] Mauro Birattari and Marco Dorigo. How to assess and report the performance of a stochastic algorithm on a benchmark problem: mean or best result on a number of runs? *Optimization Letters*, 2006.

[Chi08] Marco Chiarandini. DM811 – project description. `http://www.imada.sdu.dk/~marco/Teaching/Fall2008/DM811/Projects/project08.pdf`, 2008.

[2] Past issue: the java compiler path is `/usr/local/bin/javac`; in C, any routine that uses subroutines from the math.c library should be compiled with the `-lm` flag – eg, `cc floor.c -lm`.