PA184 – Heuristic Methods for Search and Optimisation

Assessment Details, Session Spring 2009-2010

Course instructor: Dr Dario Landa-Silva (dario.landasilva@nottingham.ac.uk)
Course organiser: Dr. Hana Rudova (hanka@fi.muni.cz)

Completion of the course requires 55 points at least
(A: 100-90, B 89-80, C 79-70, D 69-60, E59-55)

Written Examination: 70 points

- 4 questions distributed as follows:
  - 1 question on defining important concepts in heuristic search (10 points)
  - 1 question on describing structure/working of some (meta)heuristics (20 points)
  - 1 question on analysing components of evolutionary algorithms (20 points)
  - 1 question on describing/analysing constraint handling techniques (20 points)
- Duration: 1 hour
- Optional Examination Dates: **9 June 2010** and **16 June 2010**
- Exam enrollment will be open at IS MU on **10th May at 5pm**
- Resit Examination Date: **29 June 2010**
- No additional materials are allowed/required in the examination (notes, calculator, etc.)

Programming Project and Presentation: 30 points

- This project is about the design, implementation and experimental testing of heuristic and meta-heuristic methods to tackle a given combinatorial optimisation problem, namely the generalised assignment problem (GAP)
- The project is to be undertaken in teams of 3 students. Please organise yourself in teams and then a representative of the team should communicate by email to the course instructor the names of those in the team by the deadline of **7 May 2010**
- There are two deliverables:
  - Programming code and instructions
  - Presentation of work and results
- Delivery of Programming Code and Instructions
  - Single compressed file including all files that comprise your work including source, executable and a short pdf document with any instructions that are required to compile and execute your program
  - The source code must be properly documented with comments and explanations
  - Send your work by email to the course instructor and cc the course organiser by the deadline of **4 June 2010**
- Delivery of Presentation
  - 15 minute individual presentation using slides in PFD format
  - Focus on your contribution to the team's work and results in the project. Include 2 slides (1 per person) summarising the work done by the 2 other members of your team (these 2 slides don't need to be presented).
  - Delivery will be on the **same date of your examination**. Maximum 10 students will present in each optional examination date, in the 2.5 hours following the 1 hour dedicated to the examination. The presentation will be video recorded to be marked later by the course instructor. Please send your presentation slides by email to the course instructor **by 5pm on the day before your presentation**

Heuristic Approaches for the Generalised Assignment Problem

The subject problem for this coursework is the well-known Generalised Assignment Problem (GAP). The problem is described as follows.

> Given n tasks, m workers, c_{ij} is cost of assigning task i to worker j. Worker j has limited time available T_j. Worker j takes t_{ij} time to complete task i. Each task is assigned to exactly one worker. A worker can undertake more that one task depending on T_j. Assign all the tasks to the workers so that the total cost is minimised without exceeding T_j for any worker.

The mathematical programming model of this problem can be seen in the course notes. A *set of 20 problem instances is available from the course information web page* in order to conduct your computational experiments. See the OR-library for more on these test instances: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/gapinfo.html

The goal of this project is to *develop a set of heuristic approaches to tackle the GAP*, including constructive heuristics, neighbourhood search moves, meta-heuristic and any other heuristic operator that you think is necessary in order to design your main solving method.

The following steps are to be followed:

1. Select your *solution representation*, parse and load the instances data sets
2. Design and implement a *greedy constructive heuristic* for feasible solutions
3. Design and implement a *'peckish' constructive heuristic* for feasible solutions
4. Design and implement a *random initialisation heuristic* for feasible solutions
5. Devise a way to *visualise solutions and their quality* on the screen
6. Design a set of *neighbourhood moves* to generate candidate solutions
7. Design and implement a *Iterated Local Search* method
8. Design and implement the *Meta-heuristic* method assigned to your team
9. Conduct *experimental study and report results* in order to assess the performance of your heuristic methods
10. Please note that it is up to the team to decide any other aspect not specified here including constraint handling. In step 8, you are allowed to consider designing and implementing a hybrid approach with another algorithm but the meta-heuristic method assigned should play the main role in the hybrid.

The team can decide to use any of the following programming languages: Java, C/C++ or Python. Any other language must be discussed/agreed with the course instructor in advance.

The meta-heuristic methods that will be assigned randomly to the teams are immediately after the date of 7 May 2010:

1. Great deluge
2. Greedy randomised search procedure
3. Simulated annealing
4. Tabu search
5. Variable neighbourhood search

Please contact the course instructor and/or the course organiser if you have any question.