Lab sessions: a software platform for testing algorithms (TASK 2)

Fabio Caraffini

De Montfort University

School of Computer Science and Informatics
– Institute of Artificial Intelligence (IAI) –

TASK 2 –Part 1–

Implementation of Benchmark and Miscellaneous functions

Implement the following Java functions:

- open *BaseFunctions.java* and complete the code for De Jong (link), Rastrigin (link), Schwefel (link) and Michalewicz (link) problems;
- open *Misc.java* and complete the code for the two variants of "generateRandomSolution" and "toro":

```
static double[] generateRandomSolution(double[][] bounds, int n)
static double[] generateRandomSolution(double[] bounds, int n)
static double[] toro(double[] x, double[] bounds)
static double[] toro(double[] x, double[] bounds)
```

TASK 2 –Part 1–
Hints:

N.B. Assure scalability to n generic variables:

- bounds[0] is the lower bound and bounds[1] the upper bound for the variables (i.e. all of them have the same upper bound and the same lower bound);
- similarly, bounds[i][0] is the lower bound and bounds[i][1] the upper bound for the generic i-th variable.

"generateRandomSolution" is basically the generation of a random number within an interval ([lower bound, upper bound]) for each variable.

For "'toro" you can use the psuedo-code on the next slide

N.B. Comment your work and include the code in the report (or submit a zip archive containing report+code)

TASK 2 –Part 1–

Hints: pseudocode for Toroidal correction (general case with customised boudaries)

procedure toro($\mathbf{x}$)
  for $i = 0 : n$ **do**
    $\mathbf{x_{cor}}[i] \leftarrow \frac{\mathbf{x}[i]-\mathbf{x^L}[i]}{\mathbf{x^U}[i]-\mathbf{x^L}[i]}$ ▷ Normalisation
    if $\mathbf{x_{cor}}[i] > 1$ **then**
      $\mathbf{x_{cor}}[i] \leftarrow \mathbf{x_{cor}}[i] - \mathit{fix}\,(\mathbf{x_{cor}}[i])$
    else if $\mathbf{x_{cor}} < 0$ **then**
      $\mathbf{x_{cor}}\,[i] \leftarrow 1 - \left|\mathbf{x_{cor}}\,[i] - \mathit{fix}\,(\mathbf{x_{cor}}\,[i])\right|$
    end if
    $\mathbf{x_{cor}}\,[i] \leftarrow \mathbf{x^L}\,[i] + \mathbf{x_{cor}}\,[i] \cdot \left(\mathbf{x^U}[i] - \mathbf{x^L}[i]\right)$ ▷ Rescaling
  end for
  Output $\mathbf{x_{cor}}$
end procedure

N.B. X^L = lover bound vector, X^U = upper bound vector.

TASK 2 –Part 2–

- Pick one of the single-solution algorithms thought in this module (or from the literature if you like challenges) and implement it;
- make sure your code is run-able and optimise the four benchmark problems implemented in TASK 2 –Part 1– with such optimiser:
  - you can generate the initial guess with *generateRandomSolution()*;
  - embed *toro()* in the algorithm and use it when required;
- generate results and display them in the report as explained in the lectures:
  - i.e. in tables reporting avg$\pm$ std + W test !
- **Write you interpretation of the results!**

(Look at the Coursework Specification file for details, but bear in mind that the report should contain code and narrative about interpretation of results, everything else can be omitted.)

TASK 2
Further HELP & SUPPORT

- The following videos will give further explanations and guidance on how to install SOS and use it to address TASK 2
  - watching them is highly re recommended!;
- SOS/TASK 2 HOW TO:
  - VIDEO TUTORIAL 2 (CLIK HERE)
  - VIDEO TUTORIAL 2 (CLIK HERE)