MOEA - User Manual

Hamburg University of Technology
Institute of Control Systems
Andrey Popov

Contents

Introduction

Many engineering problems are characterized by several, often contradicting objectives, that have to be satisfied simultaneously. In many cases the standard approach of combining the objectives to a single one does not provide satisfactory results [1]. Evolutionary Algorithms are an attractive alternative in such cases [2], since they are a direct parallel search method and as such are able to obtain a Pareto trade-off surface between the objective function on a single run.

Strengthen Pareto Evolutionary Algorithm 2 (SPEA 2) is a Multi-Objective Evolutionary Algorithm (MOEA) developed by Computer Engineering (TIK), ETH Zurich, and reported to work well on many benchmark problems [3]. SPEA 2 is originally written in C, and developed under the PISA framework (Platform and Programming Language Independent Interface for Search Algorithms).

In order to allow easy use of SPEA 2 under MATLAB and avoid the file-based information exchange between the solver and the problem, the Institute of Control Systems, Hamburg University of Technology, combined SPEA 2 with the DTLZ variation function, modified the C files to interface with MATLAB and created an additional M file `moea.m` for user convenience.

Used abbreviations:

MOEA	Multi-Objective Evolutionary Algorithm
SPEA 2	Strengthen Pareto Evolutionary Algorithm 2
PISA	Platform and Programming Language Independent Interface for Search Algorithms

1 Copyright and Legal Notes

PISA

Platform and Programming Language Independent Interface for Search Algorithms

Computer Engineering (TIK) ETH Zurich

http://www.tik.ee.ethz.ch/pisa/

Copyright (c) 2002-2003 Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory. All rights reserved.

In no event shall the Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory be liable to any party for direct, indirect, special, incidental, or consequential damages arising out of the use of this software and its documentation, even if the Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory has been advised of the possibility of such damage.

The Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory, specifically disclaims any warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The software provided hereunder is on an "as is" basis, and the Swiss Federal Institute of Technology, Computer Engineering and Networks Laboratory has no obligation to provide maintenance, support, updates, enhancements, or modifications.

Combined MATLAB - C implementation of SPEA 2

Hamburg University of Technology, Institute of Control Systems

http://www.tu-harburg.de/rts

In no event shall the Hamburg University of Technology, Institute of Control Systems be liable to any party for direct, indirect, special, incidental, or consequential damages arising out of the use of this software and its documentation, even if the Hamburg University of Technology, Institute of Control Systems has been advised of the possibility of such damage.

Hamburg University of Technology, Institute of Control Systems, specifically disclaims any warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The software provided hereunder is on an "AS IS" basis, and the Hamburg University of Technology, Institute of Control Systems has no obligation to provide maintenance, support, updates, enhancements, or modifications.

2 Files and Options

The usage steps are described in moea_steps.pdf

spea2.dll

This is the compiled version of SPEA2 multi-objective optimization algorithm. It includes interface between MATLAB and C environment as well as the SPEA2 algorithm for selection and DTLZ functions for variation (mutation, uniform crossover and binary simulated crossover). The source codes for both selector (SPEA2) and variator (DTLZ) can be found on the PISA web-site.

The original PISA structure is removed and the selector and variator programs and combined to a single one.

The input arguments of `spea2` are similar to those, described below. To see the full list, please read the source code of `moea.m`.

The normal stopping criteria of the function is the number of generations.

To recompile `spea2` use:

```
mex spea2.c variator.c selector.c
```

moea.m

This program gets the user arguments, checks their validity and calls `spea2`, thus making the use of the program more intuitive.

The function call is in the form:

```
[genes, fitness] = moea('file_name', bounds, options)
```

`file_name` - name of the .M function, calculating the cost functions that should be minimized (!! SPEA2 always minimizes !!). The input arguments of this functions are the decision variables and the output arguments are the minimized functions.

`bounds` - $N \times 2$ matrix, containing the lower and upper boundaries for each decision variables, where N is the number of decision variables:

```
bounds = [  Lower1, Upper1;
            Lower2, Upper2;
            ...
            LowerN, UpperN];
```

Where as `LowerK` and `UpperK` mean the correspondingly the lower and upper boundaries for decision variable K.

The number of decision variables is determined by the number of rows in `bounds`.

`options` - structure, containing the optimization options. `options = moea()` loads a predefined options set. That set can be easily modified to fit the user needs, by changing the values of the options, e.g., `options.MaxGen = 100;`

`options.MaxGen` - maximal number of generations (integer > 0);

`options.PopSize` - population size (integer > 0);

`options.Dim` - problem dimension (number of objective functions to be minimized simultaneously) (integer > 0);

`options.Seed` - seed for the random generator (real);

`options.Tourn` - number of individuals for Tournament selection (integer > 1);

`options.i_mut_p` - individual mutation probability (real $\in [0, 1]$);

`options.i_rec_p` - individual recombination probability (real $\in [0, 1]$);

`options.v_mut_p` - variable mutation probability (real $\in [0, 1]$);

`options.v_swa_p` - variable swap probability (real $\in [0, 1]$);

`options.v_rec_p` - variable recombination probability (real $\in [0, 1]$);

`options.eta_mut` - eta mutation (real $\in [0, 1]$);

`options.eta_rec` - eta recombination (real $\in [0, 1]$);

`options.Display` - if different then 0, then the number of remaining generations are shown during the optimization.

For more information on the meaning of the options, please refer to the PISA documentation.

3 Usage Steps

1. Install MOEA (copy in `MATLAB\toolbox` folder and set path to it). Depending on the MATLAB version delete either `spea2.dll` (MATLAB ver. 7.1+) or `spea2.mex32` (all other). Alternatively you can re-compile SPEA2 yourself. For the purpose open the source folder in MATLAB and execute `build_spea2`.

2. Create your cost evaluation function (e.g. `myFunct.m`). The input parameter is a vector row and the function should return the results in a vector form (either row or column);

3. Define the optimization options. Call `opt = moea()` to get the standard options. Modify what needed, e.g. `opt.MaxGen = 200;`. Don't forget to specify the correct problem dimension (number of cost functions), e.g. `opt.Dim = 3;`

4. Define boundaries for each design variable. The search is constrained within the boundaries, e.g.,
   ```
   bounds = [ 0 10;   0  2;   -5  5 ];
   ```
   The number of elements of `bounds` specifies the number of decision variables

5. Start the optimization
   ```
   [genes, fitness] = moea('myFunct', bounds, opt);
   ```

References

[1] Indraneel Das and John Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems. *Structural Optimization*, 14:63–69, 1997.

[2] Carlos Manuel Mira de Fonseca. *Multiobjective Genetic Algorithms with Applications to Control Engineering Problems*. PhD thesis, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK, September 1995.

[3] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. Spea2: Improving the strength pareto evolutionary algorithm. In *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems, Proceedings of the EUROGEN2001 Conference*, 2001.