Applied Soft Computing 24 (2014) 341–362

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Physical programming for preference driven evolutionary multi-objective optimization

Gilberto Reynoso-Meza*, Javier Sanchis, Xavier Blasco, Sergio García-Nieto

Instituto Universitario de Automática e Informática Industrial, Universitat Politècnica de València, Camino de Vera s/n, Valencia 46022, Spain

ARTICLE INFO

Article history:
Received 21 November 2013
Received in revised form 27 May 2014
Accepted 5 July 2014
Available online 22 July 2014

Keywords:
Multi-objective optimization design procedure
Evolutionary multi-objective optimization
Physical programming
Many-objective optimization
Preference articulation
Decision making

ABSTRACT

Preference articulation in multi-objective optimization could be used to improve the pertinency of solutions in an approximated Pareto front. That is, computing the most interesting solutions from the designer's point of view in order to facilitate the Pareto front analysis and the selection of a design alternative. This articulation can be achieved in an *a priori*, *progressive*, or *a posteriori* manner. If it is used within an *a priori* frame, it could focus the optimization process toward the most promising areas of the Pareto front, saving computational resources and assuring a useful Pareto front approximation for the designer. In this work, a physical programming approach embedded in an evolutionary multi-objective optimization is presented as a tool for preference inclusion. The results presented and the algorithm developed validate the proposal as a potential tool for engineering design by means of evolutionary multi-objective optimization.

© 2014 Elsevier B.V. All rights reserved.

Introduction

Multi-objective optimization design (MOOD) procedures are *generate first choose later* (GFCL) holistic strategies for multi-objective problems [1]. A multi-objective problem (MOP) arises when multiple objectives and requirements must be fulfilled by the designer. Such objectives are usually in conflict with each other; therefore a trade-off solution must be calculated and selected for implementation. The GFCL strategy generates a set of potentially preferable design alternatives and the decision maker (DM or simply the designer) selects the most preferable solution according to his or her preferences. These solutions are generally Pareto optimal solutions [2].

The MOOD procedure (see Fig. 1) identifies three main (possibly fundamental) steps [3,4]: the MOP definition (measure); the multi-objective optimization process (search); and the multi-criteria decision making (MCDM) step (decision making). Major efforts are made to improve the algorithms and tools to facilitate the two latter processes.

In the case of multi-objective optimization, several algorithms have been designed (NBI [5], NNC[1] [6,7], NSGA-II[2] [8], MOGA[3] [9], MOEA/D[4] [10] for example) and used in a wide variety of applications [11–24]. Such algorithms mainly seek a set of Pareto optimal solutions that describe a Pareto front approximation. According to the designer's wishes, those algorithms would incorporate some desirable characteristics [23] such as convergence (capacity to reach the Pareto front), diversity (capacity to generate different solutions), and pertinency (capacity to generate useful solutions for the DM). For the decision making step, several tools and visualizations approaches [25] have been proposed over the years (scatter plot diagrams, parallel coordinates [26], level diagrams [27,28] or self-organizing maps [29] for example).

In [30,31] the importance of considering both processes (optimization and selection) in a holistic way, in order to guarantee a full embedment of the DM in the decision making step, was noted. This is because the decision making process is usually more time

* Corresponding author. Tel.: +34 963877007; fax: +34 963879579.
E-mail address: gilreyme@upv.es (G. Reynoso-Meza).
URL: http://cpoh.upv.es/ (G. Reynoso-Meza).

[1] Matlab code available at http://www.mathworks.com/matlabcentral/fileexchange/38976.
[2] Source code available at: http://www.iitk.ac.in/kangal/codes.shtml; also, a variant of this algorithm is available in the global optimization toolbox of Matlab.
[3] Toolbox for Matlab available at http://www.sheffield.ac.uk/acse/research/ecrg/gat.
[4] Matlab code available at http://cswww.essex.ac.uk/staff/zhang/IntrotoResearch/MOEAd.htm.

http://dx.doi.org/10.1016/j.asoc.2014.07.009
1568-4946/© 2014 Elsevier B.V. All rights reserved.

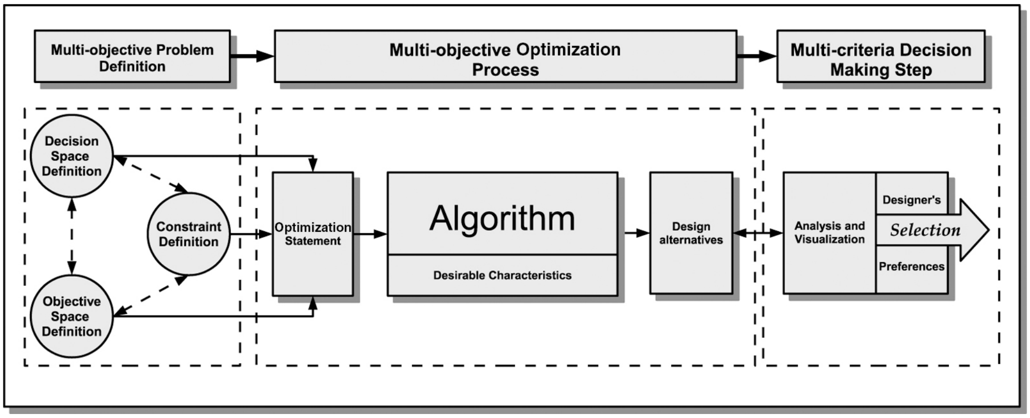

Fig. 1. A multi-objective optimization design (MOOD) procedure.

consuming than the optimization step [32]. This embedment could be achieved by providing a useful set of solutions to the designer; thereby analysing the trade-off between conflicting objectives in order to refine his or her final selection [30].

Given that the MOOD procedure should be a holistic technique, preference handling mechanisms could play a major role in bridging the gap between optimization and the selection process. These mechanisms will enable the algorithm to approximate a Pareto front with pertinent solutions in the search process; and therefore facilitating the DM's task of analysing and selecting a design alternative [33]. Furthermore, preference handling might be used in constrained optimization instances and many-objective optimization statements [34]. Challenges for preference articulation include building a practical framework to link the designer's desired trade-off with the cost function to optimize.

A first step for the aforementioned challenge, is stating meaningful design objectives. Sometimes with classical optimization approaches a cost function (or objective) is built in order to satisfy a set of requirements such as convexity and/or continuity; that is, it is built from the point of view of the optimizer despite a possible loss of interpretability. The usage of more interpretable objectives facilitates the inclusion of preferences in the optimization process, producing meaningful and pertinent solutions for the designer in the selection step. Evolutionary multi-objective optimization (EMO) provides a helpful framework for this purpose, since multi-objective evolutionary algorithms (MOEAs) have been shown to be a flexible tool to handle constrained complex functions [32] in a wide variety of engineering domain applications [11]. Furthermore, a convenient feature of using MOEAs is the possibility of selecting more interpretable objectives for the designer. That is, the objective selection could be closer to the point of view of the designer, rather than the optimizer. Nevertheless, this is just a necessary step to moving forward to preference articulation, since this could assure meaningful, but not pertinent, design alternatives.

The physical programming (PP) method [35] is very suitable for multi-objective engineering design since it formulates design objectives in an understandable and intuitive language for designers. Since it defines desirable, tolerable, and undesirable ranges for individual objectives, it becomes a potential technique to improve the pertinency of solutions in multi-objective optimization. PP has been merged previously with classical optimization techniques [1,36]; nevertheless, it remains an interesting topic to merge with MOEAs.

In this work, PP is merged with MOEAs as an auxiliary mechanism to improve the pertinency of the calculated solutions. Such an approach will enable the DM to have more useful solutions, since it provides a flexible and intuitive coding framework where the MOP is built from the DM's point of view. Although an algorithm to test its viability is developed, it could be incorporated in other MOEAs. The remainder of this work is as follows: in the section "Background" several preliminaries in multi-objective optimization, physical programming, and the MOEA are presented. In the section "Pertinency improvement by means of GPP" the preference handling mechanism is explained, and then evaluated in the section "Experimental setup". Finally, some concluding remarks are given.

Background

To state the proposal, some notions on multi-objective optimization, preference handling, physical programming, and the algorithm to be used are required. Those are provided below.

Multi-objective optimization review

As referred in [2], a MOP,[5] can be stated as follows:

$$\min_{\boldsymbol{\theta}} \boldsymbol{J}(\boldsymbol{\theta}) = [J_1(\boldsymbol{\theta}), \ldots, J_m(\boldsymbol{\theta})] \tag{1}$$

subject to

$$\boldsymbol{K}(\boldsymbol{\theta}) \leq 0 \tag{2}$$

$$\boldsymbol{L}(\boldsymbol{\theta}) = 0 \tag{3}$$

$$\underline{\theta_i} \leq \theta_i \leq \overline{\theta_i}, \quad i = [1, \ldots, n] \tag{4}$$

where $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_n]$ is defined as the decision vector with $\dim(\boldsymbol{\theta}) = n$; $\boldsymbol{J}(\boldsymbol{\theta})$ as the objective vector and $\boldsymbol{K}(\boldsymbol{\theta})$, $\boldsymbol{L}(\boldsymbol{\theta})$ as the inequality and equality constraint vectors, respectively; $\underline{\theta_i}, \overline{\theta_i}$ are the lower and upper bounds in the decision space.

It has been pointed out that there is not a single solution in MOPs, because there is not generally a better solution in all the objectives. Therefore, a set of solutions, the Pareto set, is defined. Each solution in the Pareto set defines an objective vector in the Pareto front. All the solutions in the Pareto front are a set of Pareto optimal and non-dominated solutions:

Definition 1 *(Pareto optimality [2]).* An objective vector $\boldsymbol{J}(\boldsymbol{\theta}^1)$ is Pareto optimal if there does not exist another objective vector $\boldsymbol{J}(\boldsymbol{\theta}^2)$ such that $J_i(\boldsymbol{\theta}^2) \leq J_i(\boldsymbol{\theta}^1)$ for all $i \in [1, 2, \ldots, m]$ and $J_j(\boldsymbol{\theta}^2) < J_j(\boldsymbol{\theta}^1)$ for at least one $j, j \in [1, 2, \ldots, m]$.

[5] A maximization problem can be converted to a minimization problem. For each of the objectives that have to be maximized, the transformation: $\max J_i(\boldsymbol{\theta}) = -\min(-J_i(\boldsymbol{\theta}))$ could be applied.

Download English Version:

https://daneshyari.com/en/article/6905784

Download Persian Version:

https://daneshyari.com/article/6905784

Daneshyari.com