Computers and Chemical Engineering 77 (2015) 105–115

Contents lists available at ScienceDirect

Computers and Chemical Engineering

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

Agent assisted interactive algorithm for computationally demanding multiobjective optimization problems

Vesa Ojalehto[a,*], Dmitry Podkopaev[a,b], Kaisa Miettinen[a]

[a] *University of Jyvaskyla, Department of Mathematical Information Technology, P.O. Box 35 (Agora), FI-40014 University of Jyvaskyla, Finland*
[b] *Systems Research Institute, Polish Academy of Sciences, Newelska 6, 01-447 Warsaw, Poland*

ARTICLE INFO

Article history:
Received 27 February 2014
Received in revised form
19 November 2014
Accepted 8 March 2015
Available online 28 March 2015

Keywords:
Multiple objective programming
Interactive methods
Agent-based optimization
Surrogate problem
NIMBUS
PAINT

ABSTRACT

We generalize the applicability of interactive methods for solving computationally demanding, that is, time-consuming, multiobjective optimization problems. For this purpose we propose a new agent assisted interactive algorithm. It employs a computationally inexpensive surrogate problem and four different agents that intelligently update the surrogate based on the preferences specified by a decision maker. In this way, we decrease the waiting times imposed on the decision maker during the interactive solution process and at the same time decrease the amount of preference information expected from the decision maker. The agent assisted algorithm is not specific to any interactive method or surrogate problem. As an example we implement our algorithm for the interactive NIMBUS method and the PAINT method for constructing the surrogate. This implementation was applied to support a real decision maker in solving a two-stage separation problem.

1. Introduction

In the modern society, it has become more and more important to support decision makers in finding solutions which take several conflicting objectives into account and optimize the objectives simultaneously. For such problems, it is not possible to find a single optimal solution because of the conflicting nature of the objectives. Instead of a single optimal solution, these multiobjective optimization problems have several so-called Pareto optimal solutions with different trade-offs between the objectives.

When dealing with real-world optimization problems, it is usually needed to find a single or few Pareto optimal solutions to be implemented which are called *most preferred solutions*. In order to select such a solution(s), some additional information is needed, such as how a solution should be changed in order it to get a more preferred solution for the problem, what kind of trade-offs are acceptable or what are desirable values for objective functions. This *preference information* can be obtained from a human decision maker (DM) having expertise in the problem domain. Several methods have been developed for finding the most preferable solution (see, e.g., Branke et al., 2008; Miettinen, 1999 and references therein).

In this paper, we concentrate on so-called *interactive* methods (see, e.g., Miettinen, 1999; Miettinen et al., 2008 and references therein), where the solution process makes progress iteratively by asking the DM to specify preference information until most preferred one is found. By exploring Pareto optimal solutions in this manner, the DM can learn about the trade-offs between the conflicting objective functions and, thus, gain insight about the problem. In addition, the DM can learn about how feasible his or her preferences are by comparing the expectations to the Pareto optimal solutions found. This means that the DM can even change his or her preferences during the solution process, if desired. Based on the learning the DM is able to make informed decisions on what kind of Pareto optimal solutions would best satisfy his or her preferences.

Interactive methods have given promising results for solving real-world optimization problems involving wide variety of engineering fields. These problems include optimal control of a continuous casting of steel (Miettinen, 2006, 2007), intensity modulated radiotherapy treatment planning (Ruotsalainen et al., 2009), optimizing configurations of an oxyfuel power plant process (Tveit et al., 2012), operating wastewater treatment plant (Hartikainen et al., 2015; Hakanen et al., 2013), optimal design and control of a paper mill (Steponavice et al., 2014), among others. For more examples of use of interactive methods in various fields see Ojalehto et al. (2014) and references therein.

Real-world multiobjective optimization problems can be computationally demanding. The function evaluations may depend,

* Corresponding author. Tel.: +358 408053287.
E-mail address: vesa.ojalehto@jyu.fi (V. Ojalehto).

http://dx.doi.org/10.1016/j.compchemeng.2015.03.004
0098-1354/

for example, on time-consuming computations or simulations (Hakanen et al., 2013; Hasenjäger and Sendhoff, 2005; Steponavice et al., 2014; Xu et al., 2004). If this is the case, an interactive multiobjective optimization process as outlaid above may become infeasible by the long waiting times needed to generate new Pareto optimal solutions according to the preference information specified by the DM. In other words, the interactive nature of the solution process suffers and the most preferred solutions may not be found. For example, the DM may be restricted to examining only very few Pareto optimal solutions and may stop the solution process prematurely.

One approach to solving computationally demanding problems is to replace computationally expensive functions by simplified ones. However, if the problem is simulation-based, that is, involves a simulator, it can be a so called black-box problem without any additional information about the problem besides decision variable and objective (and possibly constraint) function values. Another widely used approach is to utilize parallelization techniques to decrease the computation time. But it is possible that the problem is implemented in a way that does not allow for parallelization, e.g., the used simulator may have only a limited number of licenses available.

To summarize, when solving a computationally demanding multiobjective optimization problem using an interactive method, it is quite possible that the method requires more time to generate new Pareto optimal solutions than there is to spare. If other approaches cannot be utilized or they do not provide enough improvement in the time available, a natural way of handling such problems is to replace the computationally demanding problem with a computationally less demanding surrogate. In practice, this means that the DM is shown approximate rather than Pareto optimal solutions during the interactive solution process. However, applying the surrogate problem in multiobjective optimization has significant limitations and has been elaborated only in few studies (see e.g. Forrester et al., 2008).

A good accuracy of the surrogate problem is important in order to avoid misleading the DM. Because the preference information specified by the DM indicates what kind of solutions he or she is interested in, this information can be used to update the surrogate in an intelligent way. This means that the accuracy of the surrogate varies and is most accurate near the interesting solutions. It has been reported in the literature that solution processes with interactive methods often take quite few iterations (see e.g. Gardiner and Vanderpooten, 1997; Miettinen, 1999, pp. 134–135). One reason for this may be the cognitive load set on the DM. The load could be decreased if the amount of the preference information expected from the DM was smaller.

In this paper, we combine an interactive multiobjective objective method and a surrogate problem in an intelligent way to support the DM in order to decrease the waiting times experienced by the DM and in addition to increase the accuracy of the surrogate problem. We propose to enhance the solution process with agents, i.e., entities that try to achieve some pre-defined goals by autonomous and intelligent actions. In the proposed algorithm, we utilize the agents to update the surrogate problem near solutions that are interesting to the DM, to minimize waiting times imposed on the DM and to decrease the amount of preference information expected from the DM. We describe the proposed method as a general algorithm, as it does not depend on any specific methods or techniques. In addition to the interactive method and to the surrogate problem construction technique, the introduced agent assisted algorithm employs four different types of agents, each having their own goals.

To give more concrete ideas of how to implement agents, we demonstrate the agent assisted algorithm implemented with the classification-based NIMBUS method (Miettinen, 1999; Miettinen and Mäkelä, 2000, 2006) selected as the interactive method and the PAINT method (Hartikainen et al., 2012) selected as the surrogate problem construction technique. Furthermore, we apply the agent assisted algorithm involving the two above-mentioned methods to solve a computationally demanding two-stage separation problem and discuss the advantages achieved.

The rest of this paper is organized as follows. In Section 2, we present the concepts and background material utilized. This includes the interactive NIMBUS method and the PAINT surrogate construction technique that are used as examples. In addition, we include a brief overview of agent studies in relation to this research. We introduce the new agent assisted interactive algorithm in Section 3. In Section 4, we describe the four different agents employed by the algorithm in more detail. We demonstrate the advantages of the new algorithm by giving an example of supporting a DM in solving a multiobjective two-stage separation problem in Section 5. Finally, the paper is concluded by a discussion and concluding remarks in Sections 6 and 7, respectively.

2. Background

Next we discuss the background material used in this paper. First we briefly describe the notations used and then provide information on the methods used, that is, on the interactive NIMBUS method for multiobjective optimization and the PAINT method for constructing the surrogate problem. We finish this section by defining agents in relation to our research.

2.1. Interactive multiobjective optimization

In this paper, we consider multiobjective optimization problems of the form

$$\begin{aligned} \text{minimize or maximize} \quad & \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})\} \\ \text{subject to} \quad & \mathbf{x} \in S, \end{aligned} \tag{1}$$

where $f_i : S \to R$ are k (≥ 2) conflicting objective functions, and $\mathbf{x} = (x_1, x_2, \ldots, x_n)^T$ is the *decision (variable)* vector bounded by constraints that form the feasible set $S \subset \mathbb{R}^n$. Objective vectors $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x}))^T$ consist of *objective function* values calculated at $\mathbf{x}$.

A decision vector $\hat{\mathbf{x}}$ and the corresponding objective vector $\mathbf{f}(\hat{\mathbf{x}})$ are called Pareto optimal if there does not exist any other feasible $\mathbf{x}$ so that $f_i(\mathbf{x}) \leq f_i(\hat{\mathbf{x}})$ for all $i = 1, \ldots, k$ and $f_j(\mathbf{x}) < f_j(\hat{\mathbf{x}})$ for least one $j = 1, \ldots, k$. Such objective vectors are called Pareto optimal solutions to problem (1), and a set of Pareto optimal solutions is called a *Pareto frontier* (Miettinen, 1999). Finding the most preferred Pareto optimal solution to problem (1) is called a *solution process*. For the solution process discussed in this research, the most preferred Pareto optimal solution is found by utilizing the *DM's preferences*, i.e. information about how a solution should be changed in order to get a more preferred solution for the problem, what kind of trade-offs between objectives are acceptable for the DM or what are desirable values for objective functions.

The ranges of objective function values in the set of Pareto optimal solutions can be shown to the DM to give general understanding about attainable solutions. The k-dimensional *ideal objective vector* contains the best values of objective values whereas the worst objective function values form a *nadir objective vector*. Components of the ideal objective vector are obtained by minimizing each of the objective functions individually subject to S whereas calculating the nadir objective vector necessitates knowing the whole set of Pareto optimal solutions and thus, usually estimated values are used (for further information, see e.g. Bechikh et al., 2010; Korhonen et al., 1997; Miettinen, 1999).