Author(s): Ojalehto, Vesa; Podkopaev, Dmitry; Miettinen, Kaisa

Title: Agent assisted interactive algorithm for computationally demanding multiobjective optimization problems

Year: 2015

Version:

Please cite the original version:

Ojalehto, V., Podkopaev, D., & Miettinen, K. (2015). Agent assisted interactive algorithm for computationally demanding multiobjective optimization problems. Computers and Chemical Engineering, 77(9 June), 105-115. https://doi.org/10.1016/j.compchemeng.2015.03.004

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 (on leave)*

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., [1, 2] and references therein).

In this paper, we concentrate on so-called *interactive* methods (see, e.g., [2, 3] 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

*Corresponding author
Email address: vesa.ojalehto@jyu.fi (Vesa Ojalehto)

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 problem include optimal control of a continuous casting of steel [4, 5], intensity modulated radiotherapy treatment planning [6], optimizing configurations of an oxy-fuel power plant process [7], operating wastewater treatment plant [8, 9], optimal design and control of a paper mill [10], among others. For more examples of use of interactive methods in various fields see [11] and references therein.

Real-world multiobjective optimization problems can be computationally demanding. The function evaluations may depend, for example, on time-consuming computations or simulations [9, 12, 10, 13]. 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. [14]).

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. [15, 2, 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 [2, 16, 17] selected as the interactive method and the PAINT method [18] 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 an 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

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

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 $i = 1, ..., k$ and $f_j(\mathbf{x}) < f_j(\hat{\mathbf{x}})$ for least one $j = 1, ..., k$. Such objective vectors are called Pareto optimal solutions to problem (1), and a set of Pareto optimal solutions is called a *Pareto frontier* [2]. 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. [19, 20, 2]).

Interactive methods typically convert the original problem with the preference information specified by the DM into single objective *subproblems* [2, 3]. By selecting the subproblems well and solving them with appropriate single objective optimization methods we get Pareto optimal solutions reflecting the preferences.

The agent assisted algorithm proposed can be used with different interactive methods following the general *core structure* of interactive multiobjective optimization methods [11]. The core structure can be described as follows:

1. Initialize the process, e.g., calculate ideal and nadir objective vectors.
2. By solving a method-specific subproblem generate an initial Pareto optimal solution to be used as a starting solution.
3. Ask the DM to specify preference information related to the starting solution (in the method-specific way).

4. Generate new solution(s) based on the preference information by solving appropriate subproblem(s).

5. Ask the DM to select the most preferred solution of the previously generated solutions and denote it as the new starting solution.

6. If the selected solution is satisfactory, stop. Otherwise continue from step 3.

It should be noted that in addition to the interactive approach described here and utilized in this research, there exists several other approaches for solving multiobjective optimization problems. When classifying different approaches by the role of the DM, in addition to the *interactive methods* where the the DM's preferences are specified in an iterative process, there exist three other classes of methods [2]. If the DM's preferences are not taken available, the method is referred to as a *no preference* method. When using an *a priori* method, the DM's preferences are asked before starting the solution process. An *a posteriori* method generates a representative set of Pareto optimal solutions, that is shown to the DM. As justified in the introduction, in this research, we consider only interactive methods.

2.2. The Interactive NIMBUS Method

In this research we use the NIMBUS method [2, 16, 17] as the interactive method. The NIMBUS method is based on the classification of the objective functions. At each iteration, the DM considers the objective function values of a starting Pareto optimal solution $\mathbf{x^c}$, and is asked to classify objective functions into up to five different classes. The classes indicate what kind of changes in the objective function values would provide a more satisfactory solution than $\mathbf{x^c}$.

For simplicity, we present the classes for functions to be minimized. The classes are for functions f_i whose values

should be improved ($i \in I^<$),

should be improved to some aspiration level $\hat{z}_i < f_i(\mathbf{x^c})$ ($i \in I^\leq$),

are satisfactory at the moment ($i \in I^=$),

are allowed to impair up till some bound $\epsilon_i > f_i(\mathbf{x^c})$ ($i \in I^\geq$),

are allowed to change freely ($i \in I^\diamond$).

Based on the classification information, up to four single objective subproblems are formed. By solving these subproblems we obtain four new Pareto optimal solutions, each following the classification in a slightly different way. These solutions are shown to the DM, and he or she can select one of them or one of the previously generated Pareto optimal solutions as the most preferred solution or as a starting solution of a new classification. For a more detailed description of the NIMBUS method, see [16, 17, 11].

2.3. The PAINT Surrogate Method

In this research, by a *surrogate problem* we refer to a problem that can be used to replace the original, usually computationally expensive problem for the duration of the interactive solution process. The surrogate problem is constructed in such a way that it can be solved significantly faster than the original problem while producing optimal solutions that approximate the solutions of the original problem. Using a surrogate problem eliminates the issue of DM's waiting time during the interactive solution process but, on the other hand, poses new challenges such as controlling the accuracy.

In this research, we use the PAINT method [18] to construct a surrogate problem of a computationally demanding multiobjective optimization problem. In the PAINT method, the surrogate problem is constructed based on a precomputed set of Pareto optimal solutions. Here we refer to this set as a *constructing set*.

The constructing set can be generated with any multiobjective optimization method that generates many Pareto optimal solutions (see e.g. [21, 2, 1]). We utilize PAINT as it is applicable in both convex and nonconvex problems. More details of the PAINT method can be found in [18].

2.4. Multiagent Systems

There does not exist a single, universally agreed definition of an agent, as their usage varies from field to field. But on a general level, an *agent* is some entity, located in an some environment, where the agent tries to reach some pre-defined goal by automatic and intelligent actions [22]. Furthermore, the environment typically contains several agents interacting with each other [23]. Such an environment is called a *multiagent system*.

Agent-based computational intelligence technologies have been widely studied (see e.g. [22, 23]) and applied in many areas of science dealing with complex systems. Agent-based technologies were initially applied in information and communication, but later they have been applied in different fields related to engineering and manufacturing, such as production planning and resource allocation [24]. In addition, they have been used for single (see e.g. [25, 26, 27, 28]) and multiobjective optimization but, to our knowledge, they have not been applied in interactive multiobjective optimization discussed in this research. In [29], agents are utilized for generating Pareto optimal solutions by solving optimization problems that are similar to the subproblems used in interactive methods, but there the DM's preference information is not taken to account. In [30, 31, 32, 33, 34, 35], agents are utilized in enhancing existing evolutionary multiobjective optimization methods, where the purpose is to find a representative set of Pareto optimal solutions, that is, in an a posteriori fashion. In [36], multiple agents are utilized for supporting several DMs when solving a multiobjective optimization problem with preference-based evolutionary method. In the proposed method, agents negotiate a single reference point that should best correspond to the reference points provided by all DMs. Furthermore, in [37] agents are utilized to reduce the number of questions asked from the DM when utilizing a priori method.

In these approaches, unlike in the approach discussed in this research, the DM does not interact with the solution process in order to learn about the problem characteristics or to modify his or her preferences. Furthermore, the previous research mostly concentrates on producing either all or a representative set of Pareto optimal solutions, without discussion on how to select the most preferred Pareto optimal solution that can be the basis for practical implementation of the product or process being designed.

In our research agents directly use preference information and actively assist the DM in the interactive process of finding the most preferred solution. We define an agent to have following properties:

Emergent: agents are able to solve complex problems with a set of simple rules.

Autonomous: agents have control of their inner state and they can take actions without human intervention.

Reactive: agents take actions based on their environment.

Goal-oriented: agents aim at achieving some goal with their actions.

Communal: agents are able to communicate with other agents, be they human or artificial.

Fault tolerant: agents can attempt to recover from a failure, e.g. a failure in reaching their goal.

In the literature, it has been noted that by using multiple autonomous agents that utilize several different methods it is possible to obtain optimal solutions for complex optimization problems more efficiently in comparison to using only a single agent (see e.g. [38, 39, 40]). This effect is usually demonstrated with empirical studies, but it has been shown that the use of multiagent systems should not adversely affect convergence properties of the optimization methods [41]. Therefore we use four different agents in our algorithm.

After having defined the main concepts to be used and introduced necessary background material, in the next section we can introduce the new agent assisted algorithm.

3. Agent Assisted Interactive Multiobjective Optimization Algorithm

The aim of this research is to provide the DM with assistance when solving a computationally demanding multiobjective optimization problem with an interactive method. As mentioned earlier, interactive methods are iterative, involving the DM in each iteration. In this section we introduce an agent assisted interactive multiobjective optimization algorithm for this purpose.

3.1. Introduction to the Agent Assisted Algorithm

An interactive method shows new Pareto optimal solution(s) to the DM, who studies it/them and then specifies information on his or her preferences. Then the DM is shown new solution(s). With this iterative procedure the DM can learn about the characteristics of the problem to form a firm idea of the Pareto optimal solutions that can be attained, and which of these solutions best match with his or her preferences. At the same time the DM can adjust one's preferences. As motivated in the introduction, waiting times can become an issue with computationally demanding problems. In this section we present the background for the new algorithm which can provide new Pareto optimal solutions without waiting times.

One approach for providing the DM with new solutions in a timely manner is to replace the computationally demanding problem with a surrogate problem, as described in Section 2.3. This approach consists of the following three phases.

1. *Construction phase.* The surrogate problem is constructed.
2. *Decision phase.* The interactive method is employed to solve the surrogate problem in communication with the DM.
3. *Projection phase.* The solution of the original problem is obtained based on the solution of the surrogate problem. If needed, the surrogate problem is updated in order to improve its accuracy and the second phase is repeated.

We distinguish the decision phase where the DM is actively involved from construction and projection phases where his or her presence is not necessary. The latter two phases can be referred to as *offline phases*. By replacing the original problem with a surrogate problem we shift the computational burden from decision phase to offline phases, thus eliminating waiting time of the DM. On the other hand, this replacement means that, instead of Pareto optimal solutions the DM is shown approximate Pareto optimal solutions, i.e., Pareto optimal solutions of the surrogate problem. After the DM has found his or her most preferred approximate Pareto optimal solution, as the name of the projection phase suggests, the solution is projected to the Pareto frontier of the original problem. This can be done using, for example, an achievement scalarization function [42] as described in [11].

The challenge with a surrogate based approach is that the approximate Pareto optimal solution may be too far from the Pareto frontier of the original problem, i.e., the surrogate is not accurate enough. If the problem is computationally very demanding, the projection can take a long time. If the projected solution is too different from the corresponding approximate Pareto optimal solution, the DM may need to start the interactive solution process again. In the worst case, this may mean that first a new surrogate problem must be constructed before the interactive solution process can be started again and all previous preference information may be wasted. This outcome is the complete

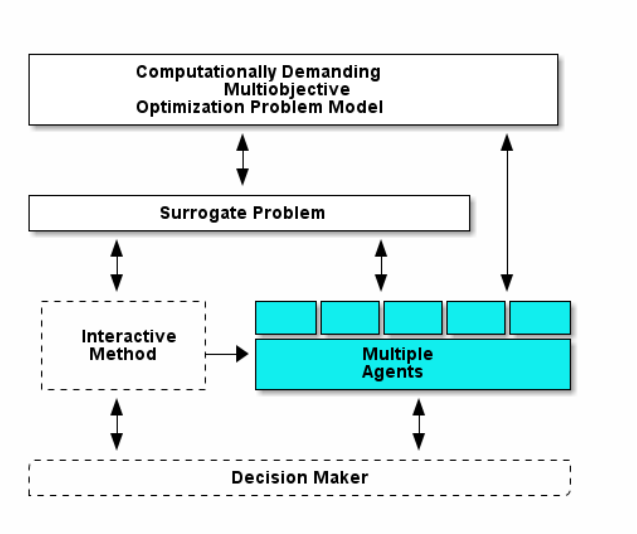

Figure 1: Overall structure of the agent assisted algorithm

opposite to the aim of using the surrogate problem because it hinders the learning process rather than supports it. Therefore, the accuracy of the surrogate problem plays an important role.

The aim of utilizing multiple independent agents, i.e., artificial decision makers, is to improve the accuracy of the surrogate problem in the intelligent way, i.e. in those areas of the problem where the improvement is most needed. We propose to utilize four different types of agents which perform specific tasks during different phases of the solution process as described in the next section.

3.2. The Agent Assisted Algorithm

Now we are in a position to describe the proposed agent assisted interactive algorithm, to be called an *agent assisted algorithm*. This algorithm is general and not tailored for any specific interactive method or surrogate problem. The agent assisted algorithm extends the interactive method by emphasizing the intelligent updating of the surrogate problem, minimizing waiting times imposed on the DM and decreasing the amount of preference information expected from the DM, thus decreasing the cognitive load.

The overall structure of elements comprising the agent assisted algorithm can be seen in Figure 1. There are four types of agents which both perform their own tasks and communicate and share information with each other. *Preference agents* use preference information expressed by the DM to build a preference model, *interactive method agents* collect information about the parameters that the interactive method uses to generate approximate Pareto optimal solutions, based on this information *optimization agents* generate new Pareto optimal

solutions, and finally, *surrogate agents* are responsible for constructing and updating the surrogate problem. Each type of agent is described in detail in the next section.

The agent assisted algorithm consists of the following six steps. We indicate for each step, to which of the phases it belongs: construction phase (c.p.), decision phase (d.p.) or projection phase (p.p.).

1. (c.p.) The surrogate agent constructs the surrogate problem
   - using information from all other agents, if available.
2. (d.p.) The DM uses the interactive method and specifies preference information based on the starting solution.
   - Preference agents collect the preference information to build a model of the DM's preferences.
3. (d.p.) The interactive method generates approximate Pareto optimal solution(s) to be shown to the DM.
   - The interactive method agents collect information on how approximate Pareto optimal solutions are generated.
4. (d.p.) The DM selects one approximate Pareto optimal solution
   (a) as the new starting solution for the next iteration and continues with step 2, or
   (b) as the most preferred solution of the surrogate problem.
   - Preference agents collect this information which is interpreted as preference of one solution over others.
5. (p.p.) The optimization agents generate new Pareto optimal solutions of the original problem based on the information collected by the interactive method agents and the preference agents.
6. The preference agents select a subset of Pareto optimal solutions which is shown to the DM. The DM either
   (a) continues with step 1, or
   (b) selects one as the most preferred solution of the original problem and stops.

The advantage of having separate offline and decision phases is that there are no waiting times for the DM to see approximate Pareto optimal solutions corresponding to his or her preferences. On the other hand, how long the offline phase can take is agreed with the DM. It should be noted that it is possible to choose one of the solutions in step 6 as the solution for the original problem using

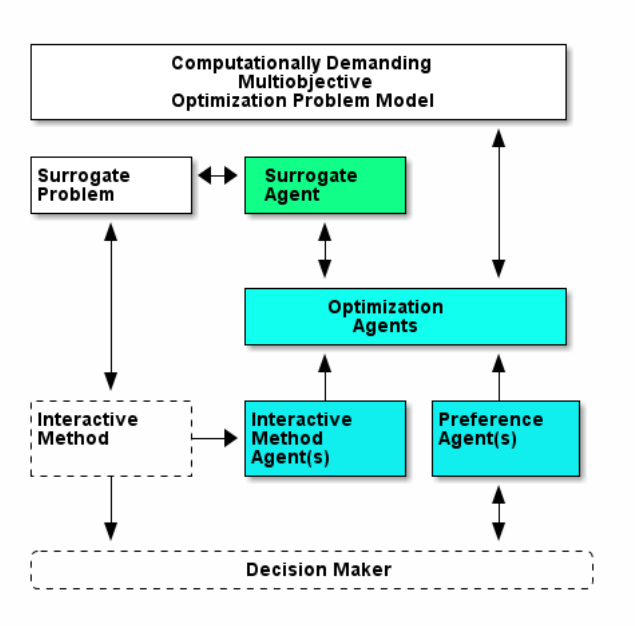

Figure 2: Detailed illustration of the agent assisted algorithm

preference agents without involvement of the DM. Therefore, in the extreme case the DM's involvement can be restricted to steps 2 to 4.

In practice, the information from all other agents in step 1 advises where the surrogate should be updated. This means that no previously specified preference information is wasted when the DM decides to continue with step 1 from step 6.

The presented description of the algorithm is very general for it can incorporate a large variety of interactive methods and surrogate problem construction techniques. To be more specific, in the next section we select both the method and the technique which allows us to describe what agents do in detail.

4. Agents in Detail

In this section we give more information about the four types of agents utilized in the agent assisted algorithm. Figure 2 provides a more detailed view of the roles of the agents in the algorithm. Because agents depend on the interactive method and the surrogate problem selected, here we provide more information assuming that NIMBUS is the interactive method and PAINT is the method to construct the surrogate problem.

In general, each agent type can be implemented in various ways, and in practice it is advisable to utilize several different agents to compliment each other. For this reason, we refer to several agents in what follows.

4.1. Preference Agents

The main function of the preference agents is to build models of the DM's preferences. These models are built in order to identify those areas of the Pareto frontier which are interesting to the DM. The model of the DM's preferences (*preference model* for short) is usually a mathematical description of all necessary information allowing one to choose a solution or to specify preference information on the DM's behalf. A preference model is generally defined as a universal rule of selecting a subset of solutions from any given set of feasible solutions, i.e., as a *choice function* [43]. Another general, but a simpler way of modeling DM's preferences is defining a *binary relation* on the set of feasible solutions (see e.g. [44, 45]) describing the DM's preference judgments for certain pairs of solutions. These two concepts are mostly used for theoretical studies. In practical multiobjective optimization methods, people use more compact and problem-specific models of DM's preferences such as value functions (see e. g. [46, 47]) and reference points (see e.g. [48, 49]).

The process of constructing a preference model based on observations of the DM's behavior is called *preference learning*. For illustrating the agent assisted algorithm, we have implemented simple techniques based on two basic approaches: *computer learning* and *human learning*. The computer learning approach is based on the assumption that all DM's input reflects his or her steady preference model and, thus, a computer learning approach can be applied for building a DM's preference model from this data. The human learning approach assumes that as the interactive method progresses, the DM learns and adjusts one's preferences. Thus constructing the preference model is reduced to predicting its parameters for the next iteration based on the time series of parameters in previous iterations. It should be noted that when using interactive methods, it cannot be assumed that the DM could take hundreds of iterations, and the selected preference model construction approach cannot depend on large amounts of input data.

As examples, we demonstrate the computer learning approach in the context of NIMBUS in two ways. Both of them are implemented using two different machine learning techniques: polynomial-based kriging (e.g., [50]) and support vector machines (e.g., [51]). In the first way, when training the agent, it is given as the input a set of (approximate) Pareto optimal solutions presented to the DM at each NIMBUS iteration, and as the output, the agent is given the solution that the DM selected from that set. After the computer learning agent has been trained with this data, it can be used to select one solution of a set of solutions (that would be most preferred by the DM). In what follows, this is referred to as a selecting agent. In the second computer learning way, the agent is given a Pareto optimal solution as the input and the NIMBUS classification specified by the DM in relation to that solution as the output. After the training, the agent will give as the output a classification information corresponding to any solution given as the input. This is called a classification agent. By combining the selecting and the classification agents, we can replace the DM in the offline phase of the agent assisted algorithm.

As an example of the human learning approach, an agent can be created for predicting coefficient of each objective function f_i in the achievement scalarizing function. Then for each agent, a feedforward multilayer neural network is trained. As the input, the agent is given the reference points (obtained from the classifications as per [17]) provided by the DM during the previous NIMBUS iterations. It is also given as the input the component of the reference point corresponding to the objective function f_i for the next classification. After the training, for each f_i, the agent can be given a reference point as the input, and it will give as the output the ith component of a new reference point. In this way, the history of the preference information specified during the NIMBUS iterations is utilized. Here preference learning can be understood as predicting how DM's input changes by analyzing the time series of previous input [52].

Besides step 2., step 6. of the agent assisted algorithm utilizes preference agents when selecting which Pareto optimal solutions should be shown to the DM. This is done by giving all Pareto optimal solutions generated as the input to the selecting agent.

4.2. Interactive Method Agents

The main function of the interactive method agents is to find Pareto optimal solutions during the offline phase. These solutions should correspond (as described below) to the approximate Pareto optimal solutions the DM has found during the decision phase. This can be achieved by first collecting information on how approximate Pareto optimal solutions were generated in the decision phase. Then, during the offline phase, this information is used to mimic the actions of the DM with the original problem to generate Pareto optimal solutions which correspond to the DM's preferences. In other words, the classifications made by the DM during the NIMBUS iterations are repeated with the original problem.

In addition, the interactive method agents can also be used for projecting approximate Pareto optimal solutions obtained in the decision phase to the Pareto frontier. Using interactive method agents in the two described ways may generate two different Pareto optimal solutions per each approximate Pareto optimal solution. Which of them is shown to the DM depends on the preference agent. This increases understanding of attainable Pareto optimal solutions but, on the other hand, also increases the computational cost. If desired, the projection can be skipped.

4.3. Optimization Agents

The main function of the optimization agent is to generate Pareto optimal solutions for the original multiobjective optimization problem. When using the NIMBUS method, Pareto optimal solutions are generated by solving single objective subproblems. In the agent assisted algorithm, these subproblems are solved by the optimization agents with several different single objective optimization methods. The methods used for solving the subproblems depend on the used interactive methods. For solving the NIMBUS subproblems, we use

global methods, such as differential evolution [53], controlled random search [54] and genetic algorithm [55], and local methods such as COBYLA [56] and proximal bundle method (if gradient information is available) [57] and their hybrids.

Let us briefly describe our approach to implementing optimization agents. When employed in the agent assisted algorithm, an optimization agent usually belongs to an *agent group*. For example, a new agent group is assembled for each step 2. taken by the DM. The goal of an agent in a group is to find a Pareto optimal solution corresponding to the preference information specified by the DM, i.e., each agent in the group tries to solve the same single objective NIMBUS subproblem. The agents in the same group differ by which single objective optimization method they employ, and by what parameters are given to those methods. In step 5, all agents are run simultaneously, but the rate of convergence for each agent in a group is studied on a decreasing interval and the agents converging fastest are given more computing time. To improve their convergence rate, the optimization agents are able to change their configuration, i.e. what method they are using and what are the method parameters. In addition, the optimization agents communicate with each other, providing information about the best solutions found and about the configurations the solutions have been found. This information is communicated also to the agents in other groups.

Optimization agents continue solving the NIMBUS subproblems until they cannot find configurations which provide improvement on the optimal values. In addition, step 5 is given a maximum time available for obtaining new Pareto optimal solution, and in practice, the optimization agents continue until the given time runs out.

As single objective optimization is not in the scope of this paper, optimization agents are not discussed here in more detail. They have been implemented following the results of [28].

4.4. Surrogate Agent

The main function of the surrogate agent is update the surrogate problem on those areas that DM has shown interest in. In the case of PAINT, it is intuitively obvious that the accuracy of the surrogate problem depends on the coverage of the constructing set. Therefore, adding a Pareto optimal solution to the set usually improves the accuracy in the approximate Pareto optimal solutions that can be obtained near that solution. Improving the accuracy can be achieved by including Pareto optimal solutions generated by optimization agents in the constructing set whenever appropriate.

If a preference agent has an inaccurate preference model, it can instruct optimization agents to generate a solution that does not correspond to the DM's preferences. Even if a surrogate agent adds it in the constructing set, the accuracy of the surrogate does not suffer because the solution added is Pareto optimal. The worst consequence of this is increased computational cost.

5. Case Study: Two-Stage Separation Problem

To demonstrate the benefits of the agent assisted algorithm we apply it in a two-stage separation problem, originally considered in [58]. The related multiobjective optimization problem is computationally demanding. The solution process was carried out with the implementations of agents, NIMBUS and PAINT contained in the IND-NIMBUS software framework where the DM used a graphical user interface. For further implementation details of the two methods, see [11, 8], respectively.

In the two-stage separation problem, an incoming feed of water and general impurity are separated into permeate and retentate. The process model considered here consists of two pumps pumping the feed to two filters and two pumps recycling a part of the permeate back to the filters. The goal of the two-stage separation problem is to extract a maximum amount of *retentate* (kg) from an incoming feed while minimizing the amount of *impurity* (kg) in the permeate and minimizing the *energy* (Kj) used by the pumps. The two-stage separation process is studied for the duration of a typical factory shift length, discretized over a time horizon. In addition to objective functions, the problem model consists of a single constraint and 100 decision variables of which four are continuous and 96 are binary valued. The problem is nonconvex, i.e., it contains multiple local optimal solutions. When using Intel® Core™ i7-2600 running at 3.4GHz, a single simulation of the problem model took 5 seconds on an average. In order to reliably obtain optimal solutions when using a differential evolution [53] method, an average of 15000 simulations were required. More details of the two-stage separation problem can be found in [58].

Originally in [58], due to time constraints, the two-stage separation problem was solved with a restricted number of function evaluations. The preferences specified by the DM and the corresponding Pareto optimal solutions generated with the interactive NIMBUS method can be seen in Table 1. Here, each section of the table represents a single iteration of the interactive method, where the first (bolded) row indicates the starting solution shown to the DM, the second row indicates the preferences specified by the DM and the following rows indicate the Pareto optimal solutions generated. For more details of the solution process, see [58].

In Table 1, each Pareto optimal solution is denoted by $\mathbf{z}_j$, of which $\mathbf{z}_7$ denotes the solution selected as the most preferred one by the DM. However, as noted in [58], the solutions generated are not actually Pareto optimal because the optimization methods did not necessarily converge, as the optimization had to be stopped prematurely due to the time constraints.

In what follows, we apply the agent assisted algorithm in the two-stage separation problem. The aim of the problem is to design a new type of separation process, and the process designer acted as the DM for during the solution process. Because we want to utilize all preference information provided in the previous research, we denote the results of [58] as the first decision phase for the agent assisted algorithm and start with the offline phase of step 5. In addition, preference agents selected only one Pareto optimal solution in step 6 with which

Iter	Issue	Max Permeate (kg)	Min Impurity	Min Energy (kJ)
	$\boldsymbol{z_1}$	**2222**	**11.02**	**16842**
1	Classif	$I^{\geq=2000}$	$I^{\leq=2.300}$	$I^{\leq=9500.000}$
	z_2^1	1732	3.92	12402
	z_3^1	1483	2.02	14632
	z_4^1	2096	3.39	16155
	$\boldsymbol{z_3^1}$	**1483**	**2.02**	**14632**
2	Classif	$I^{\leq=1900}$	$I^{\geq=2.35}$	$I^{\leq=9600.000}$
	z_5^1	950	6.84	11606
	z_6^1	1240	2.06	14939
	$\boldsymbol{z_6^1}$	**1240**	**2.06**	**14939**
3	Classif	$I^{\leq=1500}$	$I^{\geq=2.4}$	$I^{\leq=12000.000}$
	z_7^1	1348	2.14	9329
	z_8^1	1236	2.07	9339
	z_9^1	1234	1.85	9857
Pref.	$\boldsymbol{z_7^1}$	**1348**	**2.14**	**9329**

Table 1: Solution process of the two-stage separation problem in [58]

we proceeded to step 1, where the surrogate problem was constructed for the first time with the PAINT method. In this way, we could build the surrogate problem with increased accuracy on the ares that the DM had shown interested in. The DM started the second decision phase with step 2 and the agent assisted algorithm was followed till step 6 (b).

In what follows, we provide some details of the individual steps taken in the solution process. The offline phase in step 5 was started with the optimization agents first using the information from the interactive method agents to mimic the actions of the DM summarized in Table 1. For example, from the information of the iteration 3 of Table 1, four new interactive method agents were created. Each of them corresponded to one of the subproblems of the NIMBUS method generating a Pareto optimal solution (corresponding to the classification information). Each of the interactive agents employed a group of eight optimization agents. These 32 agents, using different optimization methods with different parameters, generated four new Pareto optimal solutions (one for each group). One of these solutions was $(1764, 0.20, 12030)$. To obtain this result, the optimization agents spent a total of 2600 function evaluations.

Let us consider this Pareto optimal solution for a while. It was used by the preference agent called classification agent to generate two new sets of preference information. As an example, the classification agent-based on support vector machines produced the classification $\left(I^{\leq 2270}, I^{\geq 2.3}, I^{\leq 7500}\right)$. Then this preference information was passed to interactive method agents to generate new Pareto optimal solutions. The corresponding actions were taken for each of the remaining three Pareto optimal solutions of iteration 3. Naturally, the corre-

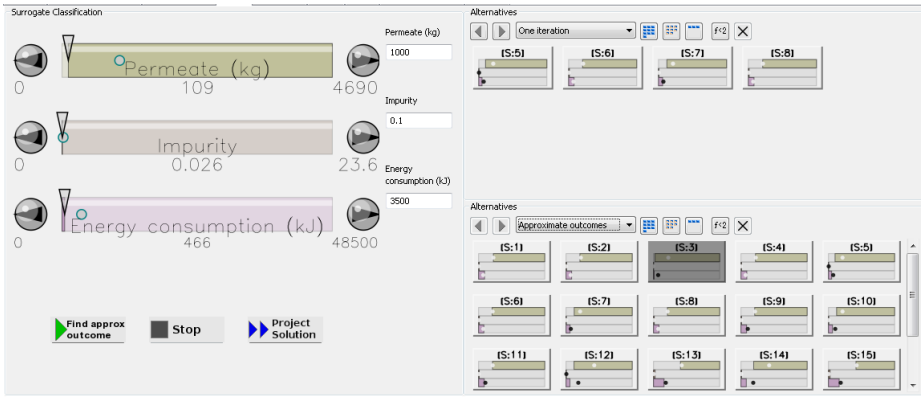

Figure 3: Graphical user interface used by the DM

sponding steps were repeated for iterations 1 and 2 of Table 1. In this way, preference information available from the previous research was utilized.

Step 5 of the first offline phase was continued until the amount of time agreed with the DM was used up. In step 6, the preference agents selected one Pareto optimal solution z_1^2 (see Table 2) which was shown to the DM. Because the DM wanted to improve it, the second offline phase was started with step 1.

In step 1, the Pareto optimal solutions generated in step 5 of the first offline phase were used to construct a surrogate problem of the two-stage separation problem. The solution process continued with the second decision phase consisting of repeated steps from 2 till 4.

A collection of the results generated during the second decision phase as well as the preference information specified can be seen in Table 2. The Pareto optimal solutions were generated based on the preferences specified by the DM in a graphical user interface shown in Figure 3. Here, the DM has provided classification for the iteration 2 of the solution process (described in Table 2), and the method has generated four new approximated Pareto optimal solutions, that are shown to the DM. In the Figure 3, the DM is shown a single Pareto optimal solution, namely the solution a_3^2, on the left side. As can be seen, the Pareto optimal solution is shown with three vertical bars, each of which corresponding to an objective function. The first objective function is to be maximized, which is indicated by having the bar starting from the right end. The lowest (estimated) value that each objective function can achieve is shown to left of the bar, and the highest (estimated) value is shown to the right. The relative position of the current objective function value is indicated with an arrow pointing down, as well as the exact numeric value below the bar. The DM indicates preferences by clicking the bar on location where the value is desired to be changed or by inputting a numeric value to the edit box located to the right of the bar. The obtained solutions are shown on the right side of the figure. For more information on the graphical user interface used during the

Iter	Issue	Max Permeate (kg)	Min Impurity	Min Energy (kJ)
	Ideal	4693	0.00	0
	Nadir	0	23.56	48532
	z_4^2	**1773**	**0.29**	**8488**
1	Classif	$I^{\geq=1050}$	$I^{<}$	$I^{\leq=3400}$
	a_1^2	1051	0.25	4380
	a_2^2	983	0.23	4100
	a_3^2	109	0.03	466
	a_4^2	1030	0.25	4294
	a_3^2	**109**	**0.03**	**466**
2	Classif	$I^{\leq=1000.0}$	$I^{\geq=0.1}$	$I^{\geq=3500.0}$
	a_5^2	469	0.10	2368
	a_6^2	962	0.29	3889
	a_7^2	580	0.12	2952
	a_8^2	988	0.28	4032
	a_5^2	**469**	**0.10**	**2368**
3	Classif	$I^{\leq=1000.0}$	$I^{\leq=0.1}$	$I^{\geq=5000.0}$
	a_9^2	974	0.23	4102
	a_{10}^2	580	0.12	2952
	a_{11}^2	990	0.23	4158
	a_5^2	**469**	**0.10**	**2368**
4	Classif	$I^{\leq=2000.0}$	$I^{\geq=0.12}$	$I^{\geq=9000.0}$
	a_{12}^2	566	0.12	2881
	a_{13}^2	1909	0.57	7735
	a_{14}^2	902	0.21	3928
	a_{15}^2	1952	0.59	7887
	a_{12}^2	**566**	**0.12**	**2881**
5	Classif	$I^{\leq=2000.0}$	$I^{\geq=0.13}$	$I^{\geq=9000.0}$
	a_{16}^2	606	0.13	3033
	a_{17}^2	1909	0.57	7735
	a_{18}^2	938	0.22	4001
	a_{19}^2	1952	0.59	7887
Pref.	a_7^2	**580**	**0.12**	**2952**

Table 2: Solution process in the second decision phase with the agent assisted algorithm

decision phase by the DM, see [11].

Based on his understating of the problem (gained during the first decision phase), the DM decided to minimize the amount of impurity as much as possible, while maintaining reasonable bounds (to be seen in the table) for the other objectives. This was the first classification. He was shown four approximate Pareto optimal solutions of which he selected a_3^2 as it had the best impurity even though the values of the other objectives were not satisfactory.

For the second iteration, he gave an upper bound 0.10 for the impurity, while aspiring for the value 1000 for the permeate (to be maximized) and maintaining an upper bound of 3500 for the energy consumption. For the third iteration, he decided to continue with a_5^2 as this solution obeyed the upper bound of impurity even though the permeate value 469 was low.

In the third iteration, he allowed increment of energy consumption till 5000, in order to obtain the desired value 1000 for the permeate. From the results shown, he noticed that by increasing the impurity level, he might be able to obtain a better permeate value while maintaining a reasonable level of energy consumption.

The fourth iteration was started again from a_5^2 to see the effect of increasing the upper bound of impurity to 0.12. As hoped for, the permeate amount improved while the impurity bound was not violated without impairing the energy consumption. He selected a_5^2.

In the fifth iteration, he allowed the impurity to rise up to 0.13 in order to improve the permeate amount. In addition, he allowed the energy consumption to rise till 9000. This iteration was not satisfactory and he selected $(580, 0.12, 2952)$ as the final approximate solution. This was the last step 4 (b) of the second decision phase.

After the second decision phase, the second offline phase of the agent assisted algorithm was started with the data summarized in Table 2. From the Pareto optimal solutions obtained in the second offline phase, the preference agents suggested four Pareto optimal solutions which would be preferred by the DM. Their number was four because it was a cognitively feasible number for the DM. These solutions are shown in Table 3. When compared to previously obtained solutions, as reported in Table 1, several of the previos solutions were worse on respect of all objective vector compontents, and therefore new solutions are strictly better than them. Of the new solutions, the DM selected z_3^3 as the most preferred solution for the two-stage separation problem. In it, the permeate and energy values were reasonable while maintaining a good impurity. The DM stopped the solution process because he was satisfied with the final solution.

As far as the solution process with the agent assisted algorithm is concerned, for the two-stage separation problem the algorithm could intelligently update the surrogate problem and enabled interaction with the DM without noticeable waiting times. To be more specific, the DM was able to obtain new approximate Pareto optimal solutions immediately after having specified preference information. This enabled him to explore the two-stage separation problem with five different preferences, that is, iteration, by spending only about half an hour of his time. This is a remarkable improvement compared to the original research

Issue	Max Permeate (kg)	Min Impurity	Min Energy(kJ)
z_1^3	323	0.10	2858
z_2^3	479	0.14	4240
z_3^3	**1870**	**0.18**	**10697**
z_4^3	2901	0.60	26348

Table 3: Pareto optimal solutions obtained from second offline phase

reported in [58], where he was able to give only one classification per day.

The construction sets generated based on the preference information specified by the DM are illustrated in Figure 4. Here, Pareto optimal solutions generated based on the first decision phase are marked with diamonds and those based on the second decision phase are marked with circles. If the agent assisted algorithm had been continued, the PAINT surrogate problem would have been constructed from a combination of these two sets. The vectors of aspiration levels specified by the DM are marked with triangles pointing downwards for the first decision phase and with triangles pointing upwards for the second decision phase. As can be seen, the agent assisted algorithm was able to identify how the DM's interests changed and to generate new solutions on the Pareto frontier in those areas.

It should be noted that the amount of preference information specified by the DM did not decrease when compared to the original research. The slowness of the original solution process was explained by the computationally demanding problem. Now that the DM could see new solutions without having to wait for hours, it encouraged him to do a more thorough search of the interesting area of the approximate Pareto frontier. Nevertheless, the preference agents decreased the cognitive load in step 6 by allowing him to consider only a subset of the generated Pareto optimal solutions.

Originally, the DM regarded impurity as the most important objective. However, during the solution process he learned that he had to trade-off and give up in impurity in order to get reasonable values for the other objectives.

The aim of this research is not simply to decrease the overall computational cost of solving a multiobjective optimization problem. The main idea is to shift the computationally demanding elements from the decision phase to the offline phase in order to decrease waiting times imposed on the DM. Interestingly, when comparing the results obtained by the optimization agents during the offline phase to the results obtained with the interactive NIMBUS method (without the surrogate problem), the optimization agents were able to converge significantly faster and, thus, computational savings were clear. For example, optimization agents were able to generate the solution z_3^1 with 1700 function evaluations, but generating a similar solution without agents took almost 15000 function evaluations. This was not studied further, as our main focus was on supporting the DM in solving multiobjective optimization problems with interactive methods.

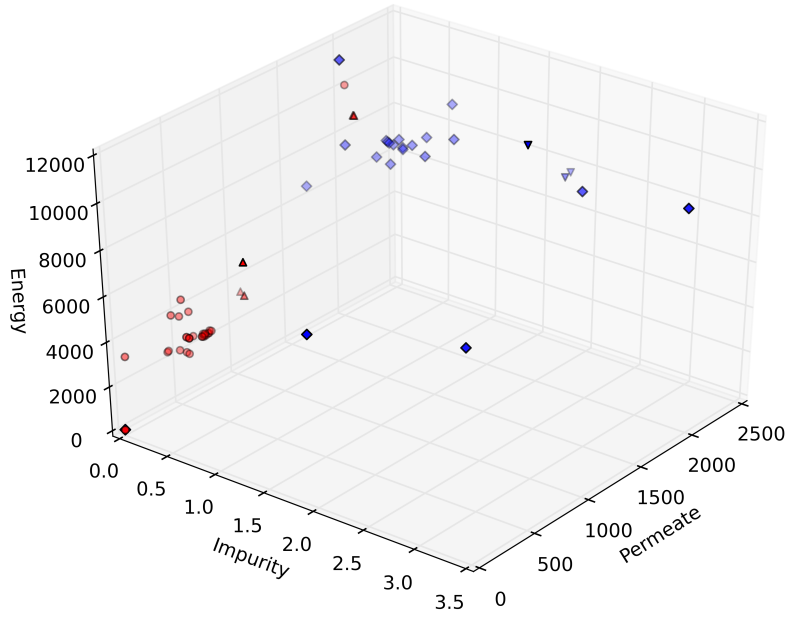

Figure 4: Construction sets created from DM's preferences

6. Discussion

With the new agent assisted algorithm we were able to generate satisfactory solutions to the DM and support him when solving the two-stage separation problem. The same DM was involved in the original research reported in [58] and found the new algorithm more convenient for exploring the Pareto frontier of the problem. He appreciated the fact that new approximate Pareto optimal solutions corresponding to his preference information were generated without waiting times.

It is intuitively evident that the agent assisted algorithm can increase the number of objective function evaluations used when solving a multiobjective optimization problem. In step 5 of the agent assisted algorithm, the agents generate more Pareto optimal solutions than the plain NIMBUS method would. However, by producing more Pareto optimal solutions based on the DM's preferences, the agent assisted algorithm is able to provide the DM with a more complete view of the problem. In this regard, the agent assisted algorithm could also be applied for computationally inexpensive problems by generating additional Pareto optimal solutions in the decision phase. The preference agents could then, as in step 6 of the agent assisted algorithm, select some of these Pareto optimal solutions to be shown to the DM. In this way, the agent assisted algorithm could encourage the DM to consider unexplored ares of the Pareto frontier that could be of interest. It could even be possible to employ the human learning agents to generate new preference information without additional function evaluations.

On the other hand, it is possible that the increase of the computational cost introduced by the agent assisted algorithm is not that significant. As pointed out in Section 5, it seems that the agent assisted algorithm was actually able to reduce the computational cost of solving the two-stage separation problem by an order of magnitude. It is possible that as the optimization agents are guided based on the preference information obtained from the DM, they are able to converge faster towards the Pareto frontier. In addition, this effect may be amplified as the optimization agents share information on the best solutions found including the history of previous iterations of the interactive method. However these results cannot be directly generalized to all solution methods.

When shortening the waiting times imposed on the DM, there is naturally some price to be paid. In the agent assisted algorithm this means showing approximate Pareto optimal solutions to the DM. This may cause some impreciseness of preference information. Clearly, the DM must be informed of this. On the other hand, real-world problems contain typically nonlinear and nonconvex objective functions. Furthermore, these problems are usually considered as a black-box, i.e. the exact formulations or even characteristics of the objectives problem are usually do not know before solving the problem. Therefore, even when solving the problem without using surrogates, we cannot guarantee that the obtained Pareto optimal solutions are accurate. In practice, the time available for solving the problem usually determines the quality of the obtained solutions. As the agent assisted algorithm allows for conducting the time inten-

sive parts of the solution process without involvement of the DM, it is possible to expand the time available and therefore provide the DM with more accurate solutions.

Naturally, if different interactive methods and surrogate problem constructing techniques are used, appropriate agents of the four types must be developed. However, the ideas of implementing agents presented here are not limited to the considered application.

7. Conclusions

In this paper, we have introduced an agent assisted interactive multiobjective optimization algorithm for solving computationally demanding problems. Motivated by the benefits of using interactive methods we, with this algorithm, enable applying them in computationally demanding problems. The algorithm is general and can be used with different interactive methods and surrogate problem construction techniques.

The algorithm employs a computationally inexpensive surrogate problem and four types of agents. The agents observe the DM's actions when using an interactive method. With these observations the agents intelligently increase the accuracy of the surrogate by updating it in the areas the DM is interested in, without a need for additional information. Besides that, the agents working with the surrogate problem shorten the waiting times imposed on the DM and decrease the amount of preference information required.

We solved a computationally demanding two-stage separation problem with the new agent assisted algorithm involving the interactive NIMBUS method and the PAINT method to construct the surrogate problem. The DM appreciated the fast solution process and the results obtained. As the experiences were most encouraging, we consider this research direction to be fruitful and conclude that agent assisted algorithm should be applied in and tested with various computationally demanding problems.

Acknowledgments

This work was supported on the part of Vesa Ojalehto by the Jyväskylä Doctoral Program in Computing and Mathematical Sciences in Finland, by the Nyyssönen foundation and by the KAUTE foundation. The authors wish to thank Mr. Jouni Savolainen for participating in this research as the decision maker.

[1] J. Branke, K. Deb, K. Miettinen, R. Slowiński (Eds.), Multiobjective Optimization: Interactive and Evolutionary Approaches, Springer-Verlag, 2008.

[2] K. Miettinen, Nonlinear Multiobjective Optimization, Kluwer Academic Publishers, 1999.

[3] K. Miettinen, F. Ruiz, A. P. Wierzbicki, Introduction to multiobjective optimization: Interactive approaches, in: J. Branke, K. Deb, K. Miettinen, R. Slowinski (Eds.), Multiobjective Optimization: Interactive and Evolutionary Approaches, Springer-Verlag, 2008, pp. 27–57.

[4] K. Miettinen, Interactive multiobjective optimization method NIMBUS applied to continuous casting of steel, in: N. Bandyopadhyay, P. Chattopadhyay, S. Chattopadhyay (Eds.), International Workshop on Neural Network and Genetic Algorithm in Materials Science and Engineering, Proceedings, Tata McGraw-Hill Publishing Company, 2006, pp. 58–72.

[5] K. Miettinen, Using interactive multiobjective optimization in continuous casting of steel, Materials and Manufacturing Processes 22 (5) (2007) 585–593.

[6] H. Ruotsalainen, E. Boman, K. Miettinen, J. Tervo, Nonlinear interactive multiobjective optimization method for radiotherapy treatment planning with Boltzmann transport equation, Contemporary Engineering Sciences 2 (9) (2009) 391–422.

[7] T. Tveit, T. Laukkanen, V. Ojalehto, K. Miettinen, C. Fogelholm, Interactive multi-objective optimisation of configurations for an oxyfuel power plant process for CO2 capture, Chemical Engineering Transactions 29 (2012) 433–438.

[8] M. Hartikainen, V. Ojalehto, K. Sahlstedt, Applying approximation method PAINT and interactive method NIMBUS to multiobjective optimization of operating a wastewater treatment plant, Engineering Optimization 47 (3) (2015) 328–346.

[9] J. Hakanen, K. Sahlstedt, K. Miettinen, Wastewater treatment plant design and operation under multiple conflicting objective functions, Environmental Modelling & Software 46 (1) (2013) 240–249.

[10] I. Steponavice, S. Ruuska, K. Miettinen, A solution process for simulation-based multiobjective design optimization with an application in paper industry, Computer-Aided Design 47 (2014) 45–58.

[11] V. Ojalehto, K. Miettinen, T. Laukkanen, Implementation aspects of interactive multiobjective optimization for modeling environments: The case of GAMS-NIMBUS, Computational Optimization and Applications 58 (3) (2014) 757–779, doi:10.1007/s10589-014-9639-y.

[12] M. Hasenjäger, B. Sendhoff, Crawling along the Pareto front: Tales from the practice, in: The 2005 IEEE Congress on Evolutionary Computation (IEEE CEC 2005), 2005, pp. 174–181.

[13] L. Xu, T. Reinikainen, W. Ren, B. P. Wang, Z. Han, D. Agonafer, A simulation-based multi-objective design optimization of electronic packages

under thermal cycling and bending, Microelectronics Reliability 44 (12) (2004) 1977–1983.

[14] A. Forrester, A. Sobester, A. Keane, Engineering Design Via Surrogate Modelling: a Practical Guide, Wiley, 2008.

[15] L. Gardiner, D. Vanderpooten, Interactive multiple criteria procedures: Some reflections, in: J. Climaco (Ed.), Multicriteria Analysis, Springer, 1997, pp. 290–301.

[16] K. Miettinen, M. M. Mäkelä, Interactive multiobjective optimization system WWW-NIMBUS on the Internet, Computers & Operations Research 27 (7-8) (2000) 709–723.

[17] K. Miettinen, M. M. Mäkelä, Synchronous approach in interactive multiobjective optimization, European Journal of Operational Research 170 (3) (2006) 909–922.

[18] M. Hartikainen, K. Miettinen, M. M. Wiecek, PAINT: Pareto front interpolation for nonlinear multiobjective optimization, Computational Optimization and Applications 52 (2012) 845–867.

[19] S. Bechikh, L. Ben Said, K. Ghedira, Estimating nadir point in multiobjective optimization using mobile reference points, in: Evolutionary Computation (CEC), 2010 IEEE Congress on, 2010, pp. 1–9.

[20] P. Korhonen, S. Salo, R. E. Steuer, A heuristic for estimating nadir criterion values in multiple objective linear programming, Operations Research 45 (5) (1997) pp.751–757.

[21] Y. Sawaragi, H. Nakayama, T. Tanino, Theory of Multiobjective Optimization, Academic Press, Inc., 1985.

[22] S. Russell, P. Norvig, Artificial Intelligence: A Modern Approach, Prentice Hall, 2003.

[23] M. Wooldridge, An Introduction to MultiAgent Systems, John Wiley & Sons, Inc., 2002.

[24] W. Shen, Q. Hao, H. J. Yoon, D. H. Norrie, Applications of agent-based systems in intelligent manufacturing: An updated review, Advanced Engineering Informatics 20 (4) (2006) 415–431.

[25] F. B. Aydemir, A. Günay, F. Öztoprak, Ş. İker Birbil, P. Yolum, Multiagent cooperation for solving global optimization problems: An extendible framework with example cooperation strategies, Journal of Global Optimization 57 (2) (2013) 499–519.

[26] I. Lobel, A. Ozdaglar, D. Feijer, Distributed multi-agent optimization with state-dependent communication, Mathematical Programming 129 (2) (2011) 255–284.

[27] R. A. Sarker, T. Ray, Agent based evolutionary approach: An introduction, in: R. A. Sarker, T. Ray (Eds.), Agent-Based Evolutionary Search, Springer Berlin Heidelberg, 2010, pp. 1–11.

[28] J. D. Siirola, S. Hauan, A. W. Westerberg, Toward agent-based process systems engineering: Proposed framework and application to non-convex optimization, Computers & Chemical Engineering 27 (12) (2003) 1801–1811.

[29] J. D. Siirola, S. Hauan, A. W. Westerberg, Computing Pareto fronts using distributed agents, Computers & Chemical Engineering 29 (1) (2004) 113–126.

[30] R. Drezewski, L. Siwik, Agent-based co-operative co-evolutionary algorithm for multi-objective optimization, in: L. Rutkowski, R. Tadeusiewicz, L. Zadeh, J. Zurada (Eds.), Artificial Intelligence and Soft Computing–ICAISC 2008, Springer Berlin Heidelberg, 2008, pp. 388–397.

[31] C. Grimme, J. Lepping, U. Schwiegelshohn, Multi-criteria scheduling: An agent-based approach for expert knowledge integration, Journal of Scheduling 16 (4) (2013) 369–383, cited By (since 1996)2.

[32] H. Li, H. Ding, Agent-based evolutionary algorithms applied to constrained multi-objective optimization problems, Applied Artificial Intelligence 26 (10) (2012) 941–951.

[33] C. Simons, I. Parmee, R. Gwynllyw, Interactive, evolutionary search in upstream object-oriented class design, Software Engineering, IEEE Transactions on 36 (6) (2010) 798–816.

[34] L. Siwik, S. Natanek, Solving constrained multi-criteria optimization tasks using elitist evolutionary multi-agent system, in: Evolutionary Computation, 2008. CEC 2008. (IEEE World Congress on Computational Intelligence). IEEE Congress on, 2008, pp. 3358–3365.

[35] K. Socha, M. Kisiel-Dorohinicki, Agent-based evolutionary multiobjective optimisation, in: Proceedings of the 2002 Congress on Evolutionary Computation, 2002. (CEC '02), Vol. 1, 2002, pp. 109–114.

[36] S. Bechikh, L. Ben Said, K. Ghedira, Negotiating decision makers' reference points for group preference-based evolutionary multi-objective optimization, in: Hybrid Intelligent Systems (HIS), 2011 11th International Conference on, 2011, pp. 377–382.

[37] D. Cvetković, I. Parmee, Agent-based support within an interactive evolutionary design system, Artificial Intelligence for Engineering Design, Analysis and Manufacturing: AIEDAM 16 (5) (2002) 331–342.

[38] P. Davidsson, J. A. Persson, J. Holmgren, On the integration of agent-based and mathematical optimization techniques, in: N. T. Nguyen, A. Grzech, R. Howlett, L. C. Jain (Eds.), Agent and Multi-Agent Systems: Technologies and Applications, Springer Berlin Heidelberg, 2007, pp. 1–10.

[39] T. Máhr, J. Srour, M. de Weerdt, R. Zuidwijk, Can agents measure up? a comparative study of an agent-based and on-line optimization approach for a drayage problem with uncertainty, Transportation Research Part C: Emerging Technologies 18 (1) (2010) 99–119.

[40] S. Talukdar, L. Baerentzen, A. Gove, P. De Souza, Asynchronous teams: Cooperation schemes for autonomous agents, Journal of Heuristics 4 (4) (1998) 295–321.

[41] J. Tsitsiklis, D. Bertsekas, M. Athans, Distributed asynchronous deterministic and stochastic gradient optimization algorithms, Automatic Control, IEEE Transactions on 31 (9) (1986) 803–812.

[42] A. Wierzbicki, A mathematical basis for satisficing decision making, Mathematical Modelling 3 (1982) 391–405.

[43] H. Uzawa, Note on preference and axioms of choice, Annals of the Institute of Statistical Mathematics 8 (1956) 35–40.

[44] K. J. Arrow, Rational choice functions and orderings, Economica 26 (102) (1959) 121–127.

[45] A. K. Sen, Choice functions and revealed preference, The Review of Economic Studies 38 (3) (1971) 307–317.

[46] E. Jacquet-Lagreze, Y. Siskos, Preference disaggregation: 20 years of MCDA experience, European Journal of Operational Research 130 (2) (2001) 233–245.

[47] S. Greco, M. Kadziński, V. Mousseau, R. Słowiński, Robust ordinal regression for multiple criteria group decision: UTAGMS-GROUP and UTADISGMS-GROUP, Decision Support Systems 52 (3) (2012) 549–561.

[48] A. P. Wierzbicki, The use of reference objectives in multiobjective optimization, in: G. Fandel, T. Gal (Eds.), Multiple Criteria Decision Making: Theory and Application, Springer, 1980, pp. 468–486.

[49] I. Kaliszewski, Out of the mist–towards decision-maker-friendly multiple criteria decision making support, European Journal of Operational Research 158 (2) (2004) 293–307.

[50] C. E. Rasmussen, C. K. I. Williams, Gaussian Processes for Machine Learning, The MIT Press, 2006.

[51] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (3) (2011) 27:1–27:27.

[52] K. Chakraborty, K. Mehrotra, C. K. Mohan, S. Ranka, Forecasting the behavior of multivariate time series using neural networks, Neural Networks 5 (6) (1992) 961–970.

[53] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (4) (1997) 341–359.

[54] W. Price, Global optimization by Controlled Random Search, Journal of Optimization Theory and Applications 40 (3) (1983) 333–348.

[55] K. Miettinen, M. M. Mäkelä, J. Toivanen, Numerical comparison of some penalty-based constraint handling techniques in genetic algorithms, Journal of Global Optimization 27 (4) (2003) 427–446.

[56] M. J. D. Powell, A direct search optimization method that models the objective and constraint functions by linear interpolation, in: S. Gomez, J. Hennart (Eds.), Advances in Optimization and Numerical Analysis, Kluwer Academic Publishers, 1994, pp. 51–67.

[57] M. M. Mäkelä, P. Neittaanmäki, Nonsmooth Optimization Analysis and Algorithms with Applications to Optimal Control, World Scientific, 1992.

[58] K. Sindhya, V. Ojalehto, J. Savolainen, H. Niemistö, J. Hakanen, K. Miettinen, Coupling dynamic simulation and interactive multiobjective optimization for complex problems: An APROS-NIMBUS case study, Expert Systems with Applications 41 (5) (2014) 2546–2558.