An Interactive Regret-Based Genetic Algorithm for Solving Multi-Objective Combinatorial Optimization Problems

Nawal Benabbou, Cassandre Leroy, Thibaut Lust

An Interactive Regret-Based Genetic Algorithm for Solving Multi-Objective Combinatorial Optimization Problems

Nawal Benabbou, Cassandre Leroy, Thibaut Lust
Sorbonne Université, CNRS, Laboratoire d'Informatique de Paris 6, LIP6 F-75005 Paris, France
nawal.benabbou@lip6.fr, cassandre.leroy@lip6.fr, thibaut.lust@lip6.fr

Abstract

We propose a new approach consisting in combining genetic algorithms and regret-based incremental preference elicitation for solving multi-objective combinatorial optimization problems with unknown preferences. For the purpose of elicitation, we assume that the decision maker's preferences can be represented by a parameterized scalarizing function but the parameters are initially not known. Instead, the parameter imprecision is progressively reduced by asking preference queries to the decision maker during the search to help identify the best solutions within a population. Our algorithm, called RIGA, can be applied to any multi-objective combinatorial optimization problem provided that the scalarizing function is linear in its parameters and that a (near-)optimal solution can be efficiently determined when preferences are known. Moreover, RIGA runs in polynomial time while asking no more than a polynomial number of queries. For the multi-objective traveling salesman problem, we provide numerical results showing its practical efficiency in terms of number of queries, computation time and gap to optimality.

Introduction

Preference-based search is an important line of research in Artificial Intelligence with various applications to investments planning, resource allocation, group configuration, recommendation systems and electronic commerce (see e.g., (Junker 2002; Auger et al. 2009; Roijers and Whiteson 2017; Marinescu, Razak, and Wilson 2017; Zintgraf et al. 2018)). The increasing complexity of applications encountered in Artificial Intelligence and other areas of Computer Science significantly complicates the task of decision makers (DM) who need to find the best solution to their problems. In many practical search problems, the evaluation of solutions involves several aspects or points of view corresponding to multiple objectives (e.g., time, cost, distance, risk). Multi-objective combinatorial optimization (MOCO) is concerned with problems involving several (conflicting) objectives/criteria to be optimized simultaneously (e.g., minimizing costs while maximizing profits).

In MOCO problems, without preference information, we only know that the best solution for the decision maker (DM) is among the Pareto-optimal solutions[1]. While the first contributions mainly focused on the generation of all Pareto optimal solutions, tackling problems with many objectives has required the development of new methods to avoid computational overhead due to the high number of Pareto optimal solutions. Many of these new contributions concern the integration of preferences. However most of them focus on the elaboration of efficient methods to identify the best solution given a preference model, considering preference elicitation as a preliminary stage. In this paper, we address both issues simultaneously. The idea is to collect preference information during the search to focus on promising solutions while reducing the elicitation burden by only asking queries to discriminate between the solutions found during the search.

Preference elicitation on combinatorial domains is an active topic that has been recently studied in various contexts, e.g. in voting problems (Benabbou and Perny 2016; Benabbou et al. 2016), in stable matching problems (Drummond and Boutilier 2014), in constraint satisfaction problems (Boutilier et al. 2006), in Markov Decision Processes (Regan and Boutilier 2011; Weng and Zanuttini 2013; Gilbert et al. 2015) and in multi-objective optimization problems (Korhonen 2005; Kaddani et al. 2017; Benabbou and Perny 2018). In this paper, we assume that the DM's preferences can be represented by a parameterized scalarizing function that is initially not known, and we study the potential of incremental elicitation (White III, Sage, and Dozono 1984) in this context. The idea is to select preference queries one at a time, to be as informative as possible, so as to progressively reduce the set of admissible preference parameters until determining a (near-)optimal solution. We focus here on regret-based incremental elicitation, an approach recently designed within the Artificial Intelligence community, as it was proved to be very effective in practice (Wang and Boutilier 2003; Boutilier et al. 2006; Benabbou and Perny 2015). More precisely, we propose a new interactive solving method that uses regret-based incremental elicitation techniques to solve hard MOCO problems with unknown preference parameters. For this purpose, we focus on genetic algorithms, which are widely applied to nu-

[1] A solution is called Pareto-optimal if no other solution is better on all objectives while being strictly better on at least one objective.

merous hard combinatorial optimization problems.

The paper is organized as follows: Firstly, we conduct a literature overview on interactive genetic algorithms and regret-based solving methods. Secondly, we recall the general principle of incremental elicitation driven by the minimax regret decision criterion. Thirdly, a new regret-based interactive genetic algorithm is proposed and illustrated on a small instance of the Multi-objective Traveling Salesman Problem (MTSP). Finally, we provide numerical tests showing its practical efficiency, comparing its performances with that of several existing methods.

Related Works

In the past decade, integrating preferences into genetic algorithms has become increasingly popular, see e.g., (Branke et al. 2008; Xin et al. 2018). Due to the high number of different interactive methods that has been developed, (Xin et al. 2018) have recently established a taxonomy identifying the important factors to differentiate these methods. Four essential design factors are defined: interaction pattern (how the interaction with the DM is scheduled during the run), preference information (how the preference information is obtained from the DM), preference model (utility function, dominance relation or decision rules), and search engine (how the interesting solutions are produced, e.g. mathematical programming techniques or heuristics).

For the new method developed in this paper, the interaction with the DM is done during the run, the preference information is retrieved from pairwise comparisons, the preference model is a utility function and the search engine can be both mathematical programming techniques or heuristics. To the best of our knowlegde, the existing methods that share the same factors are: the Interactive Evolutionary Metaheuristic (IEM) (Phelps and Köksalan 2003), the interactive Pareto Memetic Algorithm (IPMA) (Jaszkiewicz 2004), the Progressively Interactive Evolutionary Multi-Objective approach using Value Functions (PI-EMOVF) (Deb et al. 2010), the Necessary-preference-enhanced Evolutionary Multi-objective Optimizer (NEMO) (Branke et al. 2015), the Brain-Computer Evolutionary Multi-objective Optimization Algorithm (BC-EMOA) (Battiti and Passerini 2010) and the interactive evolutionary multi-objective algorithm with preference model called INSPM (Pedro and Takahashi 2014). However, all but the first two of these methods are adaptations of NSGA-II (Deb et al. 2002), which is the reference algorithm for solving continuous multi-objective problems, and they have not been tested in MOCO problems. The first two methods only consider utility functions based on linear or Chebyshev aggregation. In IEM, linear programming techniques are used to learn the parameters of the utility function, whereas an internal genetic algorithm is employed in IPMA. In (Jaszkiewicz 2004), IEM and IPMA were directly compared and IPMA achieved better results on the considered MSTP instances. IPMA follows the classical steps of genetic algorithms while generating pairwise comparison queries periodically to reduce the set of possible utility functions. The frequency of preference queries is controlled by a comparison probability, which is progressively reduced during the search. Instead, our algorithm uses regret-based incremental elicitation techniques developed in the AI community to select informative preference queries and generate promising solutions during the search. As a result, our method generates at most 26 queries on existing MTSP instances with 300 cities and 7 objectives (see the numerical section) whereas IPMA needs between 30 and 60 queries on smaller instances (150 cities and 6 objectives). Moreover, we propose to apply the crossover and mutation operators on the preference parameters instead of solutions to better cover the space of admissible utility functions.

Recently it has been proposed to combine search and regret-based incremental elicitation by asking preference queries during the construction of the (near-)optimal solution (Benabbou and Perny 2015). The basic principle consists in constructing the optimal solution from optimal subsolutions using the available preference information, asking new preference information only when necessary. In this paper, we explore another way by considering non-constructive algorithms. More precisely, we introduce an interactive genetic algorithm that uses incremental elicitation techniques to select individuals from populations. The main novelty is to make use of metaheuristics instead of constructive algorithms, which enables to tackle preference-based combinatorial optimization problems for which no efficient exact algorithm is known. Moreover, the combination of preference elicitation and genetic algorithms has several specific advantages. First, preference queries only involve complete feasible solutions. This provides at least two advantages: 1) solutions are easier to compare, and 2) no independence assumption is required in the definition of preferences. The latter point is of special interest when preferences are represented by *non-linear* decision models because the cost of partial solutions is often a poor predictor of the actual cost of their extensions. Another interest of combining regret-based incremental elicitation and genetic algorithms is to produce interactive methods with performance guarantees: the proposed method 1) generates no more than a polynomial number of queries when the population size and number of generations are polynomial (as preference information is only used to discriminate between solutions within a population), and 2) runs in polynomial time for MOCO problems that can be solved (exactly or approximately) in polynomial time when preference are known.

More recently, it has been proposed to combine search and regret based incremental elicitation in a different way (Benabbou and Lust 2019). The idea is to identify informative queries by exploiting the extreme points of the polyhedron representing the admissible preference parameters; this new method is called IEEP for Incremental Elicitation based one Extreme Points. Our algorithm and IEEP are both general in the sense that they can be applied to any MOCO problem, provided that the scalarizing function is linear in its parameters (e.g., a weighted sum, a Choquet integral) and that a (near-)optimal solution can be efficiently determined when preferences are known. However, these two methods give different performance guarantees: IEEP is an exact exponential time method that may generate an exponential number of queries, whereas our method is a poly-time heuristic that asks no more than a polynomial number of queries. These

two procedures will be compared in the numerical section.

Background and Notations

In this paper, we consider a general MOCO problem with n objective functions $y_i, i \in \{1, \ldots, n\}$, to be minimized. In this problem, any solution $x \in \mathcal{X}$ is associated with a performance vector $y(x) = (y_1(x), \ldots, y_n(x)) \in \mathbb{R}^n$, where $\mathcal{X}$ is the feasible set in the decision space and $y_i(x)$ is the evaluation of x on the i-th objective.

Here we assume that the DM's preferences over solutions can be represented by a scalarizing function f_ω that is linear in its parameters ω (e.g., weighted sums, Choquet integrals). Formally, $x \in \mathcal{X}$ is preferred to $x' \in \mathcal{X}$ iff $f_\omega(y(x)) \leq f_\omega(y(x'))$. We also assume that parameters ω are initially not known. Instead, we are given a (possibly empty) set Θ of preference statements of type $(a, b) \in \mathbb{R}^n \times \mathbb{R}^n$, meaning that the DM prefers vector a to vector b. Given Θ, we consider the set Ω_Θ of all parameters ω that are compatible with Θ the set of available preference information. Formally, we have $\Omega_\Theta = \{\omega : \forall (a, b) \in \Theta, f_\omega(a) \leq f_\omega(b)\}$. Since f_ω is linear in its parameters ω, we can assume that Ω_Θ is a convex polyhedron without loss of generality.

To make decisions with partial preference information, we use the minimax regret criterion (MMR) defined using pairwise max regrets (PMR) and max regrets (MR) as follows:

Definition 1 *For any two solutions $x, x' \in \mathcal{X}$:*
$PMR(x, x', \Omega_\Theta) = \max_{\omega \in \Omega_\Theta} \{f_\omega(y(x)) - f_\omega(y(x'))\}$
$MR(x, \mathcal{X}, \Omega_\Theta) = \max_{x' \in \mathcal{X}} PMR(x, x', \Omega_\Theta)$
$MMR(\mathcal{X}, \Omega_\Theta) = \min_{x \in \mathcal{X}} MR(x, \mathcal{X}, \Omega_\Theta)$

The set $\arg\min_{x \in \mathcal{X}} MR(x, \mathcal{X}, \Omega_\Theta)$ is the set of all optimal solutions according to the minimax regret decision criterion. Recommending any of these solutions allow to minimize the worst-case loss; in particular, these solutions are necessarily optimal according to the DM's preferences when $MMR(\mathcal{X}, \Omega_\Theta) = 0$ holds.

Note that $MMR(\mathcal{X}, \Omega_\Theta)$ can only decrease when adding new preference information in Θ. This observation has led to the following elicitation approach: reduce the parameter imprecision by iteratively asking queries to the DM until the value $MMR(\mathcal{X}, \Omega_\Theta)$ drops below a given threshold $\delta \geq 0$ representing the maximum allowable gap to optimality (Boutilier et al. 2006); if we set $\delta = 0$, then we obtain the (optimal) preferred solutions at the end of the execution. This approach, sometimes referred to as *regret-based incremental elicitation*, was efficiently used in various decision contexts, such as multicriteria decision making and voting problems (e.g., (Lu and Boutilier 2011; Benabbou, Perny, and Viappiani 2017)).

A Regret-Based Incremental Genetic Algorithm

To implement the regret-based incremental elicitation approach, one need to compute the value $MMR(\mathcal{X}, \Omega_\Theta)$ at every iteration step of the procedure. For MOCO problems, this may induce prohibitive computation times since it may require to compute the pairwise max regrets for all pairs of distinct solutions in $\mathcal{X}$ (see Definition 1). Therefore, we propose instead to combine search and regret-based incremental elicitation to reduce both computation times and number of preference queries. More precisely, we now introduce an interactive genetic algorithm that uses regret-based incremental elicitation techniques to select individuals from populations. Our algorithm, called RIGA for *Regret-based Incremental Genetic Algorithm*, follows the traditional scheme of genetic algorithms but differs in the following way:

- Our populations P are composed of pairs (ω, x_ω), where ω is a possible instance of the preference parameters and solution x_ω is f_ω-optimal (or almost).
- The crossover and mutation operators are applied on parameter instances not on solutions.

More precisely, RIGA takes as input a MOCO problem, a threshold $\delta \geq 0$, a scalarizing function f_ω with unknown parameters ω and an initial set Θ of preference statements. It proceeds as follows (see Algorithm 1):

Algorithm 1 : Regret-Based Incremental Genetic Algorithm

IN $\downarrow \delta$: threshold; Θ: a set of preference statements; f_ω: a scalarizing function with unknown parameters ω.
OUT $\uparrow$: a solution in $\mathcal{X}$.

--| Initialization of the admissible parameters:
$\Omega_\Theta \leftarrow \{\omega : \forall (a, b) \in \Theta, f_\omega(a) \leq f_\omega(b)\}$
--| Generation of the initial population:
$P \leftarrow$ `ComputeExtremePoints`(Ω_Θ)
--| Genetic Algorithm:
for 1 to M **do**
  $P \leftarrow P \cup$ `Crossover&Mutation`(P, S, μ)
  while $MMR(X_P, \Omega_\Theta) > \delta$ **do**
    --| Ask the DM to compare two solutions in X_P:
    $(x, x') \leftarrow$ `Query`(X_P)
    --| Update preference information:
    $\Theta \leftarrow \Theta \cup \{(y(x), y(x'))\}$
    $\Omega_\Theta \leftarrow \{\omega : \forall (a, b) \in \Theta, f_\omega(a) \leq f_\omega(b)\}$
  end while
  --| Move to the next population:
  $x^* \leftarrow \arg\min_{x \in X_P} MR(x, X_P, \Omega_\Theta)$
  $P \leftarrow$ `Selection`(x^*, P)
end for
return x^*

Initial Population. To generate the initial population, we have to generate a set of possible preference parameters ω and then determine a solution x_ω that is (near-)optimal for the *precise* scalarizing function f_ω using an existing solving algorithm. These parameters could be generated uniformly at random but instead we propose to use the extreme points of polyhedron Ω_Θ to better cover the feasible region.

Crossover and Mutation. As already mentioned, we perform crossovers and mutations on parameter vectors (not on solutions), and for every resulting parameter vector ω, we proceed as follows: we determine a solution x_ω that is f_ω-optimal (or almost) using an existing efficient solving

method, and then we add the pair (ω, x_ω) in population P. To obtain a population of the desired size (say S), we create new parameter vectors by means of convex combinations of parameter vectors in P. This crossover operator is of particular interest in optimization problems with imprecise preference parameters because it only generates new admissible parameters from admissible parameters. In our experiments, we create a new parameter vector ω from two parameter vectors ω', ω'' in P as follows: $\omega = \lambda\omega' + (1-\lambda)\omega''$ where $\lambda \in (0,1)$ is generated uniformly at random. Then, given a mutation rate μ, we perform Gaussian mutations on single objectives which yield very good results, but more sophisticated mutation operators would be worth investigating.

Selection. To create the new generation, we select K promising pairs from population P as follows:

- First, we determine a (near-)optimal solution in population P by means of regret-based incremental elicitation. More precisely, let X_P be the set of all solutions in P. While $MMR(X_P, \Omega_\Theta) > \delta$, the DM is asked to compare two solutions $x, x' \in X_P$ and the set Ω_Θ of admissible parameters is updated by inserting the linear constraint $f_\omega(x) \leq f_\omega(x')$ (or $f_\omega(x) \geq f_\omega(x')$ depending on her answer). Once $MMR(X_P, \Omega_\Theta)$ drops below δ, we select a solution that is optimal for the minimax regret criterion, i.e. a solution x^* in $\arg\min_{x \in X_P} MR(x, X_P, \Omega_\Theta)$.
- Then, we compute the distance in objective space between x and x^* for every pair (ω, x) in P and we select the K pairs that minimize the distance to breed the next generation. In our experiments, we use the Euclidean distance but other distances would be worth investigating.

Termination. RIGA stops after M steps and returns a solution arbitrary chosen in $\arg\min_{x \in X_P} MR(x, X_P, \Omega_\Theta)$, where P is the last generated population.

Computation Times and Number of Preference Queries. Our algorithm has different tunable parameters: S the size of the population (generated by crossovers and mutations), $K < S$ the number of pairs selected for the next generation and M the number of iterations steps. For any MOCO problem with existing solving methods that run in polynomial time (in the problem size), we can ensure that RIGA also runs in polynomial time and asks no more than a polynomial number of queries by carefully setting these tunable parameters. More precisely, if S the size of the population is also polynomial, then the selection step will generate at most S^2 comparison queries in order to determine x^* the preferred solution in the population. Then it is sufficient that M the number of iteration steps be also polynomial to obtain the desired performance guarantees. To give an example, for the MTSP with a weighted sum, we can solve PMR-optimization problems using linear programming and apply the Lin-Kerninghan heuristic (Lin and Kernighan 1973) to efficiently solve the corresponding MOCO problem with precise preferences.

For illustration purposes, we now present an execution of our algorithm on a small MTSP instance:

Example 1 *Consider the instance of the MTSP depicted in Figure 1, which includes* 6 *vertices and* $n = 3$ *additive cost functions to be minimized. In this optimization problem, the set* $\mathcal{X}$ *of feasible solutions is the set of all Hamiltonian cycles, i.e. cycles that include every node exactly once. We assume here that the DM's preferences over Hamiltonian cycles can be represented by a weighted sum with the hidden weight* $\omega^* = (0.2, 0.1, 0.7)$. *We now apply RIGA on this instance with* $\delta = 0$, $S = 4$, $K = 2$, $M = 2$.

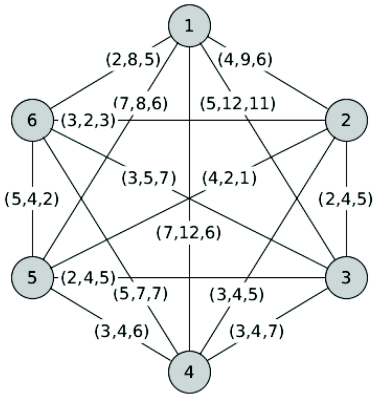

Figure 1: A MTSP instance with 6 vertices and 3 objectives.

We start the procedure with an empty set of preference statements (i.e. $\Theta = \emptyset$*); hence* Ω_Θ *the set of preference parameters is formally defined by:*

$$\Omega_\Theta = \{\omega = (\omega_1, \omega_2, \omega_3) \in (0,1)^3 \mid \omega_1 + \omega_2 + \omega_3 = 1\}.$$

In Figure 2, polyhedron Ω_Θ *is represented by the triangle ABC in the space* (ω_1, ω_2)*;* ω_3 *is implicitly defined by the constraint* $\omega_3 = 1 - \omega_1 - \omega_2$.

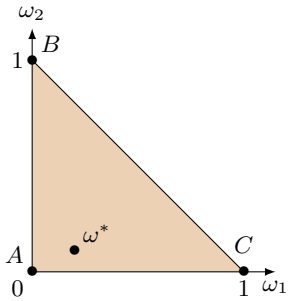

Figure 2: Initial set Ω_Θ.

Initial population: *We generate the initial population by determining one optimal solution for every extreme point of polyhedron* Ω_Θ. *For this problem, we have* $P = \{(\omega^A, x^A), (\omega^B, x^B), (\omega^C, x^C)\}$ *where* $\omega^A = (0,0,1)$, $\omega^B = (0,1,0)$, $\omega^C = (1,0,0)$, $y(x^A) = (23, 34, 26)$, $y(x^B) = (20, 30, 31)$, *and* $y(x^C) = (17, 34, 34)$.

First iteration step: *First, we must perform crossovers and mutations to obtain a population of size $S = 4$; since $|P| = 3$, we have to generate one more pair. Assume that vector $\omega' = (0.05, 0.45, 0.5)$ is generated by RIGA by applying the crossover operator on $\omega^B = (0, 1, 0)$ and $\omega^A = (0, 0, 1)$ and then performing a Gaussian mutation on the first objective. Then, weighted sum $f_{\omega'}$ is optimized by RIGA, resulting in the generation of solution x' whose cost vector is $(21, 32, 27)$ and the insertion of (ω', x') in P.*

Now the selection step begins. We must ask preference queries to the DM until $MMR(X_P, \Omega_\Theta) \leq \delta = 0$, where $X_P = (x^A, x^B, x^C, x')$. Since $MMR(X_P, \Omega_\Theta) = 4 > 0$, we ask the DM to compare two solutions in X_P, say x^A and x^B. Since $f_{\omega^*}(y(x^B)) = 28.7 > f_{\omega^*}(y(x^A)) = 26.2$, the DM answers: "solution x^A is better than solution x^B". Then Θ the set of preference statements is updated by adding the pair $(y(x^A), y(x^B))$; thus we have $\Theta = \{((23, 34, 26), (20, 30, 31))\}$. Moreover, Ω_Θ the set of admissible parameters is restricted by imposing the linear constraint $f_\omega(y(x^A)) \leq f_\omega(y(x^B))$, i.e. $\omega_2 \leq \frac{5}{9} - \frac{8}{9}\omega_1$. Now Ω_Θ is represented by the triangle ADE in Figure 3.

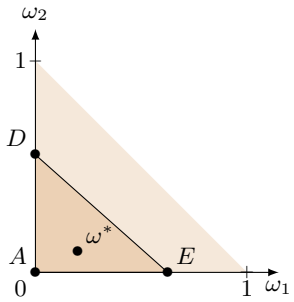

Figure 3: Ω_Θ after 1 query.

Since we have $MMR(X_P, \Omega_\Theta) = 0.75 > 0$, the DM is asked again to compare two solutions in X_P, say x^A and x^C. Here the DM prefers solution x^A to solution x^C since we have $f_{\omega^*}(y(x^C)) = 30.6 > f_{\omega^*}(y(x^A)) = 26.2$. Therefore we must perform the following updates: we add the pair $((23, 34, 26), (20, 30, 31))$ to Θ and we restrict Ω_Θ by imposing the linear constraint $f_\omega(y(x^A)) \leq f_\omega(y(x^C))$, i.e. $\omega_2 \leq 1 - \frac{7}{4}\omega_1$ (see Figure 4 where polyhedron Ω_Θ is represented by ADFG).

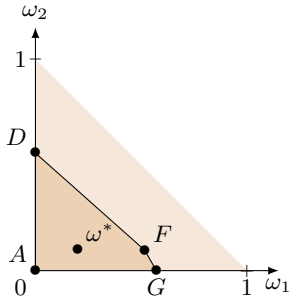

Figure 4: Ω_Θ after 2 queries.

Now we have $MMR(X_P, \Omega_\Theta) = MR(x^A, X_P, \Omega_\Theta) = 0 \leq \delta$. We move to the next population: $x^* = x^A$ and as $K = 2$ we need to select one more pair. The closest solution according to the Euclidean distance is x' and thus we have $P = \{(\omega^A, x^a), (\omega', x')\}$.

Second iteration step: *Since $|P| = 2$ and $S = 4$, we need to generate two more pairs. Assume that we obtain $(0.08, 0.4, 0.52)$ and $(0.01, 0.1, 0.89)$ by applying the crossover operator on ω^A and ω', and then performing a mutation on the resulting vectors. We then optimize the corresponding weighted sums and we obtain solutions x'' and x''' whose cost vectors are $y(x'') = (21, 32, 27)$ and $y(x''') = (23, 34, 26)$. Hence we have $P = ((\omega^A, x^A), (\omega', x'), (\omega'', x''), (\omega''', x'''))$. Since $MMR(X_P, \Omega_\Theta) = MR(x^A, X_P, \Omega_\Theta) = 0 \leq \delta$, no questions are needed at this step. This step ends by setting $x^* = x^a$ and $P((\omega^A, x^A), (\omega''', x'''))$.*

Thus, after two generations, RIGA stops by returning the solution $x^* = x^A$ (corresponding to cycle $1 - 4 - 3 - 2 - 5 - 6 - 1$) which is actually the optimal solution according to the DM's preferences. Note that only two preference queries were needed to discriminate between the 60 feasible solutions (among which 10 are Pareto-optimal).

Experimental Results

In this section, we provide numerical results aiming to estimate the performance of RIGA in terms of computation time (given in seconds), number of queries and gap to optimality (expressed in terms of percentage from the optimal solution). In our experiments, we use existing Euclidean instances of the MTSP with 50 and 300 cities, and $n = 3$ to 7 objectives[2].

The DM's Preferences. We first assume that the DM's preferences over solutions can be represented by a weighted sum f_ω with imprecise weights, and we start with an empty set of preference statements; therefore we initially have $\Omega_\Theta = \{\omega \in \mathbb{R}^n_+ : \sum_{i=1}^n \omega_i = 1\}$. During the execution of RIGA, the answers to queries are simulated using a weighting vector ω randomly generated before running the algorithm. This weighting vector is generated using the procedure presented in (Rubinstein 1982) so as to guarantee a uniform distribution of the weights.

Query Generation Strategy. Recall that, at each iteration step of RIGA algorithm, we ask preference queries to the DM until the inequality $MMR(X_P, \Omega_\Theta) \leq \delta$ holds, where X_P is the set of all solutions in the current population P. Here queries are generated using the Current Solution Strategy (Boutilier et al. 2006): at each step, the DM is asked to compare a solution $x \in X_P$ achieving the minimax regret to one of its adversary's choice (i.e. a solution in $\arg\max_{y \in X_P} PMR(x, y, \Omega_\Theta)$).

[2] https://eden.dei.uc.pt/~paquete/tsp/

Implementation Details. Numerical tests were performed on a Intel Core i7-8550U CPU with 16 GB of RAM, with a program written in C++. PMR-optimizations are performed by CPLEX Optimizer[3] and MTSP instances with precise weights are solved using the Lin-Kernighan heuristic from the Concorde website[4]. To generate the initial population, the library polymake[5] is used to determine the extreme points of polyhedron Ω_Θ.

In this section, we will compare the following regret-based solving procedures:

- RIGA: the proposed algorithm. Due to space constraint, we only give the results obtained with $M = 15$ generations, $S = 30$ solutions per iteration, $K = 5$ solutions selected per iteration, and $\delta = 0.1$, which corresponds to the best results achieved by RIGA on these instances.
- RIGA$_S$: RIGA where genetic operators are directly applied to solutions (instead of parameter vectors). Here we use single point crossovers and Gaussian mutations on single objectives.
- IEEP$_\delta$: the exact solving method introduced in (Benabbou and Lust 2019) which returns a solution $x \in \mathcal{X}$ such that $\text{MR}(x, \mathcal{X}, \Omega_\Theta) \leq \delta$ holds at the end of the procedure.
- Two-Phase$_\delta$: a two-phase method which consists in reducing the size of the Pareto front by constructing a "well-represented" set (Jaszkiewicz 2018) and then applying the CSS strategy on this set until the minimax regret drops below threshold $\delta \geq 0$; here we generate sets of size 3000.

In the following tables, the results are averaged over 100 runs and "/" means that the timeout (45 mins) is exceeded.

Firstly, we compare the performances of RIGA and RIGA$_S$ (see Tables 1-2). We observe that RIGA$_S$ generates very few preference queries (at most 5 against 26 for RIGA) and ends after only one minute (against three minutes for RIGA). This shows that RIGA$_S$ mainly produces solutions that are quite easy to discriminate without preference information (Pareto-dominated solutions); note that RIGA cannot produce Pareto-dominated solutions when using an exact solving method to generate new solutions during crossovers and mutations. Moreover, RIGA$_S$ induces an error that is much larger than that of RIGA (about 20% against at most 2%). Therefore, we can conclude that RIGA generates solutions that are more relevant according to the DM's maker preferences. Thus it seems more appropriate to apply genetic operators on possible parameters rather than applying them on feasible solutions for solving MOCO problems with unknown preference parameters.

Secondly, we compare RIGA to IEEP$_\delta$. We use two different tolerance thresholds: $\delta = 0$ to obtain the optimal solution and $\delta = 0.02$ to allow for the same error as RIGA (to be as fair as possible). In Table 3, we observe that computing the optimal solution is very costly both in terms of computation time (the timeout is exceeded for $n > 5$) and number of preference queries (56 queries are needed for $n = 5$). Then, when comparing Tables 1 and 4, we see that IEEP$_{0.02}$ is better than RIGA on the smaller instances ($n = 3$ criteria) but RIGA outperforms IEEP$_{0.02}$ on the bigger instances; to give an example, for $n = 7$ criteria, RIGA is significantly faster than IEEP$_{0.02}$ (12 times faster) and both procedures need no more than 26 queries to solve the problem.

n	time(s)	queries	error
3	130.80	17.92	0.04
4	171.14	20.10	0.22
5	178.23	21.88	0.62
6	185.27	24.06	1.12
7	202.23	25.54	1.97

Table 1: RIGA for the MTSP instances with 300 cities.

n	time(s)	queries	error
3	53.08	2.01	18.78
4	62.41	3.58	19.63
5	65.21	3.76	21.29
6	65.75	4.13	19.47
7	67.50	4.89	21.09

Table 2: RIGA$_S$ for MTSP instances with 300 cities.

n	time(s)	queries	error
3	156.76	19.77	0
4	445.69	34.00	0
5	1817.68	56.30	0
6	/	/	/
7	/	/	/

Table 3: IEEP$_0$ for MTSP instances with 300 cities.

n	time(s)	queries	error
3	50.46	6.47	1.07
4	132.02	10.47	1.16
5	307.87	15.53	0.91
6	802.49	19.80	1.06
7	2336.73	25.67	0.94

Table 4: IEEP$_{0.02}$ for MTSP instances with 300 cities.

Thirdly, we compare RIGA to Two-Phase$_\delta$ with $\delta = 0.02$ (see Tables 1 and 5). In all the settings, RIGA outperforms Two-Phase$_\delta$ in terms of computation time, number of queries and error[6]. This illustrates the efficiency of interweaving search and regret-based elicitation for the determination of a near-optimal solution in combinatorial domains.

[3] https://www.ibm.com/analytics/cplex-optimizer

[4] http://www.math.uwaterloo.ca/tsp/concorde

[5] http://polymake.org

[6] Note that the error with TwoPhase$_\delta$ may exceed 2% since it focuses on a subset of the Pareto front.

n	time(s)	queries	error
3	182.17	11.55	0.67
4	346.44	18.38	1.16
5	422.79	24.16	1.48
6	542.82	31.86	2.78
7	546.18	36.02	3.78

Table 5: Two-Phase$_{0.02}$ for MTSP instances with 300 cities.

Finally, we give the results obtained by RIGA when assuming that the DM's preferences are represented by an ordered weighted averaging (OWA) operator (Yager 1988):

$$f_\omega(x) = \sum_{i=1}^{n} \omega_i y_{(i)}(x)$$

where $\omega = (\omega_1, \ldots, \omega_n) \in \mathbb{R}^n_+$ and $(.)$ is a permutation sorting the components of $y(x) = (y_1(x), \ldots, y_n(x))$ in decreasing order. Moreover, we use decreasing weights to model preferences for well-balanced solutions (as the larger weights are associated with the worst values). Since the OWA operator is not linear in $y_{(x)}$, we cannot reduce the multi-objective TSP to a single-objective TSP. Therefore, to approximately solve the problem with known weights, we use a simple local search based on the 2-opt neighborhood function (Croes 1958). In addition, to solve the problem with known weights exactly (to be able to compute the error), we use a well-known linearization of the OWA operator with decreasing weights (see (Ogryczak and Sliwinski 2003)). Since solving the corresponding linear program is very costly in practice (more than two hours for 50 cities and $n = 6$), we only provide the results for the MTSP instances with 50 cities and n ranging from 3 to 6 objectives. Results are averaged over 100 runs.

In Table 6, we observe that RIGA is very efficient both in terms of computation times and number of queries. Moreover, we see that the error is below 1% for all the instances. This shows that RIGA can also be used to solve multi-objective optimization problems with complex decision models, even for problems such that there is no efficient algorithm for the determination of the optimal solution with known preference parameters.

n	time(s)	queries	error
3	42.92	15.50	0.89
4	39.26	15.80	0.65
5	40.91	15.47	0.83
6	44.13	15.63	0.88

Table 6: RIGA for MTSP instances with 50 cities and an OWA aggregation function.

Conclusion

In this paper, we have proposed a general method based on genetic algorithms and regret-based preference elicitation (called RIGA) for solving hard multi-objective combinatorial optimization problems with imprecise preference parameters. Our interactive method makes use of regret-based incremental elicitation techniques to generate informative preference queries during the search. Moreover, one can ensure that RIGA asks no more than a polynomial number of queries (in the problem size) by simply limiting the size of the populations and the number of generations. Similarly, one can ensure that RIGA runs in polynomial time for MOCO problems that can be (approximately) solved in polynomial time when preferences are precisely known.

We have applied the proposed method on existing instances of the MTSP, considering two different aggregation functions: the weighted sum and the ordered weighted averaging operator. We have compared the performances achieved by RIGA on these instances with that of different regret-based interactive solving methods. In particular, the provided numerical results show that our method achieves better results than IEEP (the state-of-the-art regret-based incremental solving method for the MTSP) on the bigger instances (300 cities and more than 4 criteria). Overall, RIGA was shown to be very efficient in practice: it returned very high quality solutions (the error was always less than 2%) in a few minutes with a relatively small number of queries.

Our algorithm can be applied to any MOCO problem for which preferences are represented by an aggregation function that is linear in its parameters. The next step is to extend our approach to more complex decision models (e.g., Chebyshev functions) so as to enhance descriptive and perspective possibilities.

Acknowledgements

This work was supported by the Paris Ile-de-France Region.

References

[Auger et al. 2009] Auger, A.; Bader, J.; Brockhoff, D.; and Zitzler, E. 2009. Articulating user preferences in many-objective problems by sampling the weighted hypervolume. In *GECCO '09*, 555–562.

[Battiti and Passerini 2010] Battiti, R., and Passerini, A. 2010. Brain-computer evolutionary multiobjective optimization: A genetic algorithm adapting to the decision maker. *IEEE Trans. Evolutionary Computation* 14(5):671–687.

[Benabbou and Lust 2019] Benabbou, N., and Lust, T. 2019. A general interactive approach for solving multi-objective combinatorial optimization problems with imprecise preferences. In *SOCS'19*, 164–165.

[Benabbou and Perny 2015] Benabbou, N., and Perny, P. 2015. Incremental weight elicitation for multiobjective state space search. In *Proceedings of AAAI'15*, 1093–1098.

[Benabbou and Perny 2016] Benabbou, N., and Perny, P. 2016. Solving multi-agent knapsack problems using incremental approval voting. In *ECAI'16*, 1318–1326.

[Benabbou and Perny 2018] Benabbou, N., and Perny, P. 2018. Interactive resolution of multiobjective combinatorial optimization problems by incremental elicitation of criteria weights. *EURO journal on decision processes* 6:283–319.

[Benabbou et al. 2016] Benabbou, N.; Di Sabatino Di Diodoro, S.; Perny, P.; and Viappiani, P. 2016. Incremental preference elicitation in multi-attribute domains for choice and ranking with the Borda count. In *SUM'16*, 81–95.

[Benabbou, Perny, and Viappiani 2017] Benabbou, N.; Perny, P.; and Viappiani, P. 2017. Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence* 246:152180.

[Boutilier et al. 2006] Boutilier, C.; Patrascu, R.; Poupart, P.; and Schuurmans, D. 2006. Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence* 170(8–9):686–713.

[Branke et al. 2008] Branke, J.; Deb, K.; Miettinen, K.; and Slowiński, R., eds. 2008. *Multiobjective Optimization: Interactive and Evolutionary Approaches*. Berlin, Heidelberg: Springer-Verlag.

[Branke et al. 2015] Branke, J.; Greco, S.; Sowiski, R.; and Zielniewicz, P. 2015. Learning value functions in interactive evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation* 19(1):88–102.

[Croes 1958] Croes, G. 1958. A method for solving traveling-salesman problems. *Operations research* 6(6):791–812.

[Deb et al. 2002] Deb, K.; Pratap, A.; Agarwal, S.; and Meyarivan, T. 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Trans. Evol. Comp* 6(2):182–197.

[Deb et al. 2010] Deb, K.; Sinha, A.; Korhonen, P. J.; and Wallenius, J. 2010. An interactive evolutionary multiobjective optimization method based on progressively approximated value functions. *IEEE Transactions on Evolutionary Computation* 14(5):723–739.

[Drummond and Boutilier 2014] Drummond, J., and Boutilier, C. 2014. Preference elicitation and interview minimization in stable matchings. In *AAAI'14*, 645–653.

[Gilbert et al. 2015] Gilbert, H.; Spanjaard, O.; Viappiani, P.; and Weng, P. 2015. Reducing the number of queries in interactive value iteration. In *ADT'15*, 139–152.

[Jaszkiewicz 2004] Jaszkiewicz, A. 2004. Interactive multiobjective optimization with the pareto memetic algorithm. *Foundations of Computing and Decision Sciences* 32:15–32.

[Jaszkiewicz 2018] Jaszkiewicz, A. 2018. Many-objective Pareto local search. *European Journal of Operational Research* 271(3):1001–1013.

[Junker 2002] Junker, U. 2002. Preference-based search and multi-criteria optimization. In *Proceedings of the Fourteenth Conference on Innovative Applications of Artificial Intelligence, Edmonton, Alberta, Canada.*, 34–40.

[Kaddani et al. 2017] Kaddani, S.; Vanderpooten, D.; Vanpeperstraete, J.-M.; and Aissi, H. 2017. Weighted sum model with partial preference information: application to multi-objective optimization. *EJOR* 260:665–679.

[Korhonen 2005] Korhonen, P. 2005. *Multiple criteria decision analysis*. Greco, Salvatore and Figueira, J and Ehrgott, M. Springer.

[Lin and Kernighan 1973] Lin, S., and Kernighan, B. W. 1973. An effective heuristic algorithm for the traveling-salesman problem. *Oper. Res.* 21(2):498–516.

[Lu and Boutilier 2011] Lu, T., and Boutilier, C. 2011. Robust approximation and incremental elicitation in voting protocols. In *IJCAI'11*.

[Marinescu, Razak, and Wilson 2017] Marinescu, R.; Razak, A.; and Wilson, N. 2017. Multi-objective influence diagrams with possibly optimal policies. In *AAAI'17*, 3783–3789.

[Ogryczak and Sliwinski 2003] Ogryczak, W., and Sliwinski, T. 2003. On solving linear programs with the ordered weighted averaging objective. *European Journal of Operational Research* 148(1):80 – 91.

[Pedro and Takahashi 2014] Pedro, L. R., and Takahashi, R. H. 2014. INSPM: An interactive evolutionary multiobjective algorithm with preference model. *Information Sciences* 268:202 – 219.

[Phelps and Köksalan 2003] Phelps, S. P., and Köksalan, M. 2003. An interactive evolutionary metaheuristic for multiobjective combinatorial optimization. *Management Science* 49(12):1726–1738.

[White III, Sage, and Dozono 1984] White III, C. C.; Sage, A. P.; and Dozono, S. 1984. A model of multiattribute decisionmaking and trade-off weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics* 14(2):223–229.

[Regan and Boutilier 2011] Regan, K., and Boutilier, C. 2011. Eliciting additive reward functions for markov decision processes. In *IJCAI'11*, 2159–2164.

[Roijers and Whiteson 2017] Roijers, D. M., and Whiteson, S. 2017. Multi-objective decision making. *Synthesis Lectures on Artificial Intelligence and Machine Learning* 11(1):1–129.

[Rubinstein 1982] Rubinstein, R. 1982. Generating random vectors uniformly distributed inside and on the surface of different regions. *European Journal of Operational Research* 10(2):205 – 209.

[Wang and Boutilier 2003] Wang, T., and Boutilier, C. 2003. Incremental Utility Elicitation with the Minimax Regret Decision Criterion. In *IJCAI'03*, 309–316.

[Weng and Zanuttini 2013] Weng, P., and Zanuttini, B. 2013. Interactive value iteration for markov decision processes with unknown rewards. In *IJCAI'13*, 2415–2421.

[Xin et al. 2018] Xin, B.; Chen, L.; Chen, J.; Ishibuchi, H.; Hirota, K.; and Liu, B. 2018. Interactive multiobjective optimization: A review of the state-of-the-art. *IEEE Access* 6:41256–41279.

[Yager 1988] Yager, R. R. 1988. On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Trans. Syst. Man Cybern.* 18(1):183–190.

[Zintgraf et al. 2018] Zintgraf, L. M.; Roijers, D. M.; Linders, S.; Jonker, C. M.; and Nowé, A. 2018. Ordered preference elicitation strategies for supporting multi-objective decision making. In *AAMAS'18*, 1477–1485.