Journal Pre-proof

Multiobjective Combinatorial Optimization with Interactive Evolutionary Algorithms: the case of facility location problems

Maria Barbati, Salvatore Corrente, Salvatore Greco

PII: S2193-9438(24)00003-7
DOI: https://doi.org/10.1016/j.ejdp.2024.100047
Reference: EJDP 100047

To appear in: *EURO Journal on Decision Processes*

Received date: 17 January 2023
Revised date: 23 January 2024
Accepted date: 15 February 2024

Please cite this article as: Maria Barbati, Salvatore Corrente, Salvatore Greco, Multiobjective Combinatorial Optimization with Interactive Evolutionary Algorithms: the case of facility location problems, *EURO Journal on Decision Processes* (2024), doi: https://doi.org/10.1016/j.ejdp.2024.100047

Multiobjective Combinatorial Optimization with Interactive Evolutionary Algorithms: the case of facility location problems

Maria Barbati[a], Salvatore Corrente[b], Salvatore Greco[b]

[a] *Ca' Foscari University of Venice, Department of Economics, Cannaregio 873 Fondamenta San Giobbe, 30121 Venice, Italy*
[b] *Department of Economics and Business, University of Catania, Corso Italia, 55, 95129 Catania, Italy*

Abstract: We consider multiobjective combinatorial optimization problems handled by preference-driven efficient heuristics. They look for the most preferred part of the Pareto front based on some preferences expressed by the user during the process. In general, the Pareto set of efficient solutions is searched for in this case. However, obtaining the Pareto set does not solve the decision problem since one or more solutions, being the most preferred for the user, have to be selected. Therefore, it is necessary to elicit their preferences. What we are proposing can be seen as one of the first structured methodologies in facility location problems to search for optimal solutions taking into account the preferences of the user. To this aim, we use an interactive evolutionary multiobjective optimization procedure called NEMO-II-Ch. It is applied to a real-world multiobjective location problem with many users and many facilities to be located. Several simulations have been performed. The results obtained by NEMO-II-Ch are compared with those obtained by three algorithms knowing the user's "true" value function that is, instead, unknown to NEMO-II-Ch. They show that in many cases, NEMO-II-Ch finds the best subset of locations more quickly than the methods knowing the whole user's true preferences.

Keywords: Multiobjective Optimization, Combinatorial Optimization, Preferences, NEMO, Facility Location problems

1. Introduction

Multiple Objective Combinatorial Optimization (MOCO) problems (for a survey, see [30]) are very complex and challenging to solve. They can be approached with different methodological approaches, but, in general, one focuses on the computation of all the efficient solutions (see [69] for a discussion on the different concepts of solutions of a MOCO problem). In general, the number of efficient solutions grows exponentially with the size of the problem [30]. The combination of this, along with the inherent complexity associated with the "non-smoothness" of the optimization problems, requires a significant computational undertaking that far surpasses the effort needed for handling single objective cases. [2]. The high number of efficient solutions and the required very high computational effort are considered the main bottleneck of the MOCO problem [2, 19]. These considerations have led to the creation of several strategies that adopt heuristics to determine an approximation of the whole set of nondominated or efficient solutions, requiring less computational effort than exact algorithms. [31]. Nevertheless, it should be noted that while these may be considered the primary challenging factors

Email addresses: maria.barbati@unive.it (Maria Barbati), salvatore.corrente@unict.it (Salvatore Corrente), salgreco@unict.it (Salvatore Greco)

of a MOCO problem in theory, there are additional complexities to consider when applying it in real-life situations. It would be difficult to claim that a problem has been solved, even if a comprehensive list of efficient solutions has been generated. This list may consist of numerous solutions, potentially even in the thousands, leaving the Decision Maker (DM) feeling overwhelmed when selecting one or more. [2, 19]. Hence, aside from the computational constraints, there are several practical issues to consider regarding the support provided to the DM. With this in mind, the algorithms can leverage the integration of the DM's expressed preferences to direct the search towards the most appealing part of the Pareto front for them. Considering different moments in which the DM is asked to provide their preferences, in the literature, one distinguishes between a priori, interactive, and a posteriori methods [30]:

- in *a priori* methods, the preferences of the DM are articulated at the beginning of the process [e.g. 14],
- in *interactive* methods, the DM expresses their preferences during the search [e.g. 78],
- in *a posteriori* methods, the DM is given a set of all efficient solutions, and these solutions are subsequently examined based on their preferences [e.g. 20].

On the one hand, using a priori methods asks the DM to define their preferences that are translated by some particular utility function at the beginning of the procedure. This assumes that the DM is rational and their decisions are driven by existing preferences that just need to be uncovered. However, this assumption is not always valid as the DM may be uncertain about their preferences at the start and may need to develop them throughout the decision-making process [64, 65].
On the other hand, in the a posteriori methods, the DM is often presented with many solutions. This approach has some drawbacks too, since:

- the DM has to choose the best solution(s) analyzing the tradeoffs among objectives [19],
- presenting the complete range of solutions can overwhelm the DM, posing challenges in selecting the most optimal one(s). [48].

Based on our previous discussion, it seems that interactive methods are the most suitable approach [18, 69, 73]. Consequently, for MOCO problems, a reasonable strategy is to utilize specific heuristics that focus on discovering some efficient solutions that are the most preferred by the DM. This means that the heuristics used to explore the feasible solution set should integrate preference information provided by the DM, directing the search towards certain areas of the Pareto front that contain the most favored solutions for the DM. This is achievable by using recently suggested heuristics [9] that combine both the ability to explore feasible solutions efficiently (common in optimization-oriented heuristics like NSGA-II [23] or SPEA [79]) and the ability to construct a decision model that reflects the preferences of the decision maker (typical in some approaches of Multiple Criteria Decision Aiding (MCDA), such as ordinal regression [40, 46]). An example of such a combined methodology is the recently introduced NEMO-II-Ch algorithm [10], which incorporates the search procedure of NSGA-II along with the preference representation obtained from nonadditive robust ordinal regression [3]. This approach, which drives the search for optimal solutions guided by a preference model incorporating the preferences expressed by the DM, seems to be a very promising approach to MOCO problems in real-life applications. Indeed, it can give appropriate answers to all the challenges of MOCO problems that we have described:

- it handles the large number of efficient solutions of a MOCO problem by looking only at small subsets of efficient solutions that the DM appreciates,

- it manages the computational workload by employing proven heuristics that are highly efficient in addressing intricate multiobjective problems,
- it handles the request for decision support by driving the whole search algorithm by the preferences step by step expressed by the DM in an interactive procedure.

To test the usefulness of such an approach in this paper, we consider a typical MOCO problem, i.e. a Facility Location Problem (FLP) [34].

In FLPs we aim to locate a set of facilities in a space, optimizing some objective functions and satisfying some constraints. Historically, FLPs have been modeled using a mono-objective approach in which a single objective function has been adopted. Many contributions have been proposed in this sense with a multitude of objectives adopted [35] for describing several very different applications [51]. In reality, though, DMs must manage multiple conflicting objectives all at once. Therefore, the algorithms that they use ought to take into account a multi-objective formulation of the given problem. [30].

The classical approach for selecting the location of a facility involves defining a function that relates the distances between the potential facility users and the facility itself. [33]. The objective function becomes a linear mathematical expression of the distances to optimize. By adding multiple constraints, a combinatorial optimization model is created, and the optimal solution can be determined by solving the model, as described in what is considered the seminal paper by [41]. Therefore, the main aim becomes the theoretical development and the description of the properties of the models and their solutions [51]. Some reviews gather the basic knowledge on location science as in [60]. Similarly, the main developments in the field are analyzed in the recent books of [29] and [51].

Multiple Objective Facility Location Problems (MOFLPs) have captured the attention of researchers, especially in the last decade. Many objectives can be used, from the classical distance-related objectives to the environmental and ecological criteria (for a list, see [34]). Most of the methodologies aim to find the whole Pareto front or a part of it, implying a considerable computational effort [2]. To this aim, several methodologies can be adopted: from exact approaches (e.g., [42, 58]) to multiobjective evolutionary algorithms (e.g.[22]) for complex problems.

The underlying belief in all these approaches is that the DM possesses the capability to choose the optimal option for themselves. This assumption is based on the presumption that the DM has distinct and well-established preferences and acts entirely rationally. In most practical problems, these assumptions are not very realistic [2, 54]. Moreover, very few papers directly consider the opinion of the DMs. Frequently, the stated objectives are derived from factors related to the specific problem, without delving deeper into the perspectives of DMs. In light of this, handling complex MOFLPs with an optimization algorithm guided by DM's preferences seems an exciting approach to be explored. From this viewpoint, our contribution can be seen as the first comprehensive methodology for MOFLPs that seeks optimal solutions while taking into account the preferences of the user. For this reason, we propose to deal with MOFLPs by using NEMO-II-Ch. Interactively, the DM specifies their preferences on some pairs of possible facility locations assignments. This process directs the search towards the most appealing section of the Pareto front based on the DM's preferences, and prevents the exploration of solutions that do not meet their expectations, thereby saving time [28].

To highlight the effectiveness of NEMO-II-Ch in solving MOCO problems, we conducted simulations using varying user value functions. We then compared the performance of the algorithm with three other algorithms, namely EA-UVF [10], EA-UVF1 and EA-UVF2, which rely on the user's "true" value function. We observed that NEMO-II-Ch often performs better than the algorithms knowing the user's preferences. In order to examine the impact of the DM's preference information and cognitive burden on the convergence of NEMO-II-Ch to the preferred solution, we explored three variants. These variants involved asking the DM to compare one pair of solutions every 5, 10, and 20 generations, respectively. The results proved that asking for preference information parsimoniously is better than requiring an unrealistic cognitive effort from the user. Therefore, this sheds light on the necessity to carefully study how often the user should be queried with a pairwise comparison of solutions to ensure and speed the convergence of the algorithm [52].

The paper is structured as follows. In Section 2, an overview of location problems is provided; MCDA and, in particular, NEMO-II-Ch are presented in Section 3; the particular MOLFP to which we applied NEMO-II-Ch is described in Section 4, while the three algorithms based on the complete knowledge of the user's preference with which NEMO-II-Ch is compared are presented in Section 5. The experimental setup and the numerical results are detailed in Section 6; in Section 7, we discuss the obtained results; finally, the last section provides some conclusions together with possible avenues of research.

2. Review on recent approaches to location problems

According to [35], three types of objectives can be adopted when locating facilities. The $mini-max$ problems, known as center problems, aim to minimize the maximum distance between a user and its assigned facility [21]. Several variants of the center problems can be identified (see e.g. [12]). For instance, recently, [68] proposed a new formulation to address a situation where the k-th largest weighted distance between the users and the facilities must be minimized. The $mini-sum$ problems minimize the sum of the distances between users and facilities; this objective is well-known and much studied and called the median problem [55]. Among the median problems let us recall the Discrete Ordered Median Problem (DOMP) [26], where the objective is the minimization of an ordered weighted average of the distances of the users to the facilities. Therefore, in this variant of the problem, each user can be seen as an objective. Lastly, the *covering models* aim to find solutions in which the maximum number of users is covered, i.e., users are positioned within a given threshold distance from a facility [5].

MOFLPs have been generated from these classical location problems optimizing at the same time more objectives. The very first example was proposed by [71], which optimized the median and the center objectives. Indeed, it proposes to use the median together with other objectives. [13] proposed a multiobjective model in which the classical median problem is integrated with a robustness measure that considers potential demand changes. [7] adopted as an additional objective the maximization of the distance from the nearest affected region to decrease the impact of the facility on the population. On a similar topic, [63] described a model in which the total number of users affected by the facility is minimized. [47] modified the median problem in the presence of more DMs, considering that the evaluation of the distances between users and facilities is different for each DM. Finally, covering objectives are combined with median objectives as in [58].

Minimizing the distances between the facility (e.g. a disposal site) and the users is also defined in [19]. They generate solutions containing one of the two objectives adopted (minimizing the distance from the container) and imposing a threshold distance that counts as user dissatisfaction.

In addition to the described objectives, equality measures can be adopted as objective functions in FLPs [53]. These measures are often combined with an efficient objective (e.g., median) to avoid inefficient solutions far from all the users [33]. For example, [59] minimized the sum of the absolute differences, the equality measure, and the sum of squared users-facility distances, either to be minimized or maximized for a desirable or obnoxious facility, respectively.

[28] included all the different types of objectives described so far. They model how to choose the location for a given number of casualty collection points in the State of California. They adopt five objectives: the median, the center objective, the covering objectives (using two different distance thresholds), and the variance as an equality measure.

It can be noted that many models also include location costs that can depend on several parameters for different potential locations, such as construction costs or maintenance costs [50]. Other

MOFLPs adopting several objectives can be found in the recent survey by [34], often related to the particular case study. They also categorized the MOFLPs based on the developed methodology, identifying both exact and heuristic approaches [52]. Beyond that, several metaheuristics have been applied. Among these, we focus our attention on evolutionary algorithms [77]. The first group of applications uses NSGA-II [23]. For example, in [75] NSGA-II is adopted for the choice of the location of depots in the Colombian coffee supply network, maximizing the cover provided by the depots, minimizing the costs of locating the depots, and minimizing the distances from purchasing centers to the depot. Similarly, in [6], the NSGA-II methodology is implemented for the location of warehouses and distribution centers in the supply chain perspective, optimizing the cost of locating warehouses and the cost of transportation from these. Another case for the location of the warehouses in the supply chain is reported in [70]. In addition to that, some specific applications are approached in [25] for the location of public services in high-risk tsunami areas or in [44] for the selection of the best raster points in a Geographical Information System. Finally, [62] proposed a generic problem in which the first objective function minimizes the total setup cost of facilities while the second minimizes the total expected traveling and waiting time for the customers.

Other examples of evolutionary algorithms include the application of SPEA2 in [43] for deciding the location of depots that serve a single product type to several customers. Furthermore, Swarm Optimization has been used in [76] for approximating the Pareto front in a bi-objective FLP.

While several applications are tackled with evolutionary algorithms, very few examples have been proposed in the literature implementing interactive methods [31]. A first example of the use of interactive evolutionary multiobjective techniques for MOCO problems was proposed by [61]; the authors presented an interactive methodology for two well-known MOCO problems where the DM's utility function is estimated by comparing population members. An offspring with a better-estimated utility function triggers a new interaction with the DM. In addition, thanks to a probabilistic evaluation, the interactions can be proposed for some offspring with a lower utility function, especially at the beginning of the process, to ensure that the algorithm is not overly assured relying on the estimated utility function.

In [57], for some generic objective functions of the distances between users and the facility, the DM is asked to indicate some reference levels to be introduced as constraints in the model. Many years later, [48] proposed for the two objectives $mini - max$ and $mini - sum$ an interactive geometrical branch and bound algorithm in which good regions for the location of the facility are selected through the interaction with the DM. In [24], a memetic algorithm integrates DM's preferences. In particular, the DM can indicate reference levels for the objectives, or they can provide the upper bound on the objective function levels. The algorithm can be adapted for several MOFLPs. Besides, [37] proposed an EMO algorithm introducing convex preference cones to guide the selection of the solutions to the most preferred ones of the DM. In this sense, they introduced the possibility of introducing DM preferences in an EMO algorithm for MOCO problems. Later, in [52], the authors proposed an adaption of the same algorithm for some MOCO problems, including two MOFLPs with two or three objectives. By examining multiple factors, such as the number of interactions, the design of their interaction with the DM, and the inconsistencies in their judgment, they found that the results are quite resilient. Alterations to these parameters do not have an impact on the quality of the solutions obtained.

A useful tool to help DMs in the interactive phase can be the use of the Geographical Information System (GIS) to help DMs visualize the potential solutions as in [1]. Recently, [36] developed a Decision Support System for a bi-objective problem; in the computation phase, the lexicographic optima and the ideal point are found, while in the dialog phase, the DM can choose the area in which looking for more non-dominated solutions, analyzing maps provided in a GIS environment. This process can be repeated until the DM is satisfied with the final position for the facilities [2]. Lately, evolutionary methods in a GIS environment have also been adopted by [4] to deal with a zoning management problem for marine spatial planning.

3. Brief Introduction to MCDA and NEMO-II-Ch

3.1. MCDA and the Choquet integral

Evolutionary multiobjective optimization techniques, as noted in earlier sections, are effective in solving intricate multiobjective optimization problems. Using these algorithms will provide the user with a collection of possibly optimal solutions that are well-distributed across the Pareto front. The user is, therefore, asked to choose among them the best one(s) with respect to their preferences. Making this choice could be challenging as there tends to be a considerable number of non-dominated solutions, which might make the DM feel uneasy about making a selection.

To avoid this, in recent years, interactive methods have been spread out [8]. They aim to include some preference information from the part of the DM addressing the search to the subset of the Pareto front more interesting for them. To do that, MCDA methods are used together with evolutionary algorithms (for an updated state-of-the-art survey on MCDA see [40]).

Given a set of alternatives $A = \{a, b, \ldots\}$ evaluated on a set of n evaluation criteria $G = \{f_1, \ldots, f_n\}$[1], MCDA methods deal with ranking, choice, and sorting problems. We will be more interested in ranking and choice problems in this case. In ranking problems, all considered alternatives have to be rank-ordered from the best to the worst, while, in choice problems, the best alternative (eventually more than one) has to be chosen, removing all the others. Since the dominance relation[2] stemming from evaluating the alternatives on the criteria at hand is too poor, several aggregation methods can be considered. In this paper, we will use as an aggregation method the Choquet integral [15] (see [38] for a survey on the use of the Choquet integral in MCDA), a method that can be included under the family of Multiattribute Value Theory (MAVT) [49]. MAVT methods are based on value functions $U : A \rightarrow \mathbb{R}$ such that the greater the value assigned to an alternative a by U, that is $U(a)$, the better a can be considered. In particular, a preference ($\succ$) and an indifference ($\sim$) relations can be defined such that $a \succ b$ iff $U(a) > U(b)$, while $a \sim b$ iff $U(a) = U(b)$.

The most common value function U is the additive one:

$$U(a) = U(f_1(a), \ldots, f_n(a)) = \sum_{j=1}^{n} u_j(f_j(a)), \tag{1}$$

where, $u_j : A \rightarrow \mathbb{R}$ are non-decreasing functions of the evaluations $f_j(a)$ for all $f_j \in G$. Moreover, due to its simplicity, the additive value function most used in applications is the weighted sum

$$U(a) = U(f_1(a), \ldots, f_n(a)) = WS(a) = \sum_{j=1}^{n} w_j \cdot f_j(a) \tag{2}$$

where w_j are the weights attached to criteria $f_j \in G$ such that $w_j \geqslant 0$ for all $f_j \in G$ and $\displaystyle\sum_{j=1}^{n} w_j = 1$. Using an additive value function assumes that the set of criteria is mutually preferentially independent [49] even if, in real-world applications, this assumption is not always verified. Indeed, the evaluation criteria can present a certain degree of positive or negative interaction. On the one hand, two criteria positively interact if the weight assigned to them (together) is greater than the sum of the weights

[1] Let us observe that the criteria in MCDA will be the objective functions of the considered multiobjective optimization problem on which the different solutions have to be evaluated.

[2] An alternative a dominates an alternative b iff a is at least as good as b for all considered criteria and better for at least one of them.

assigned to the two criteria taken alone. On the other hand, two criteria are negatively interacting if the weight assigned to them (together) is lower than the sum of the weights assigned to the two criteria singularly. Non-additive integrals are used in literature to address the interaction between criteria [38]; among them, the most well-known is the Choquet integral [15].

The Choquet integral is based on a set function $\mu : 2^G \to [0,1]$ (referred to as capacity) satisfying the following constraints:

1a) $\mu(\emptyset) = 0$ and $\mu(G) = 1$ (normalization),

2a) $\mu(S) \leqslant \mu(T)$ for all $S \subseteq T \subseteq G$ (monotonicity).

Let us observe that a capacity is a function assigning a value not only to every single criterion but to all possible subsets $S \subseteq G$ of criteria. $\mu(S)$ represents the weight of the set of criteria S and this is not necessarily equal to the sum of the weights of single criteria in the set, i.e., we could have

$$\mu(S) \neq \sum_{f_i \in S} \mu\left(\{f_i\}\right).$$

Capacities permit the representation of possible interactions between the criteria mentioned above.

Given $a \in A$, the Choquet integral of $(f_1(a), \ldots, f_n(a))$ with respect to μ (in the following, for the sake of simplicity, we shall write "the Choquet integral of a w.r.t. μ) is computed as follows

$$C_\mu(a) = C_\mu(f_1(a), \ldots, f_n(a)) = \sum_{j=1}^{n} \left[f_{(j)}(a) - f_{(j-1)}(a)\right] \mu(\{f_i \in G : f_i(a) \geqslant f_{(j)}(a)\}) \tag{3}$$

where $((1),(2),\ldots,(n))$ is a permutation of the indices of criteria $(1,2,\ldots,n)$ such that $0 = f_{(0)}(a) \leqslant f_{(1)}(a) \leqslant \cdots \leqslant f_{(n)}(a)$.

Let us observe that the original formulation of the Choquet integral is the following:

$$C_\mu(a) = \int_{0=f_{(0)}(a)}^{f_{(n)}(a)} \mu\left(\{f_i \in G : f_i(a) \geqslant t\}\right) dt = \sum_{j=1}^{n} \int_{f_{(j-1)}(a)}^{f_{(j)}(a)} \mu\left(\{f_i \in G : f_i(a) \geqslant t\}\right) dt. \tag{4}$$

For each $j \in \{1,\ldots,n\}$ and for each $t \in \left]f_{(j-1)}(a), f_{(j)}(a)\right]$,

$$\mu\left(\{f_i \in G : f_i(a) \geqslant t\}\right) = \mu\left\{f_i \in G : f_i(a) \geqslant f_{(j)}(a)\right\}$$

so that

$$\int_{f_{(j-1)}(a)}^{f_{(j)}(a)} \mu\left(\{f_i \in G : f_i(a) \geqslant t\}\right) dt = \left(f_{(j)}(a) - f_{(j-1)}(a)\right) \mu\left\{f_i \in G : f_i(a) \geqslant f_{(j)}(a)\right\}$$

and, therefore, eq. (4) boils down to eq. (3). This explains the reason for which eq. (3) represents an integral.

The ability of the Choquet integral to represent the possible interactions between criteria is nevertheless counterbalanced by its complexity since $2^{|G|}$ values (one for each possible subset of criteria in G) need to be defined. The Möbius transformation of the capacity μ can be utilized, along with k-additive capacities, to consider the interaction between criteria while typically requiring fewer parameters. This approach tends to be more efficient in terms of the number of parameters needed. [39]. Formally,

- the Möbius transformation of the capacity μ is a set function $m : 2^G \to \mathbb{R}$ such that $\mu(S) = \sum_{T \subseteq S} m(T)$ for all $S \subseteq G$ (conversely, $m(S) = \sum_{T \subseteq S} (-1)^{|S-T|}\mu(T)$ for all $S \subseteq G$) and constraints 1a) and 2a) are replaced by the following ones:

  1b) $m(\emptyset) = 0$, $\sum_{T \subseteq G} m(T) = 1$,

  2b) for all $f_j \in G$ and for all $S \subseteq G \setminus \{f_j\}$, $\sum_{T \subseteq S} m(T \cup \{f_j\}) \geqslant 0$.

  In this case, the Choquet integral of a w.r.t. μ can be written as follows:

$$C_\mu(a) = C_\mu(f_1(a), \ldots, f_n(a)) = \sum_{T \subseteq G} m(T) \min_{f_j \in T} f_j(a); \tag{5}$$

- a capacity μ is said k-additive if its Möbius transformation m is such that $m(T) = 0$ for all $T \subseteq G$ such that $|T| > k$.

The use of k-additive capacities involves the definition of $1 + n + \binom{n}{2} + \cdots + \binom{n}{k}$ coefficients $m(T)$. with $|T| \leqslant k$. In the following, we shall consider and apply the Choquet integral using a 2-additive capacity since it involves the definition of $n + \binom{n}{2}$ parameters only (one parameter $m(\{f_j\})$ for each $f_j \in G$ and one parameter $m(\{f_i, f_j\})$ for each $\{f_i, f_j\} \subseteq G$) and it is generally sufficient in practice to represent the preferences of the DM [39].

By using a 2-additive capacity, the Choquet integral can be written in the following way

$$C_\mu(a) = C_\mu(f_1(a), \ldots, f_n(a)) = \sum_{f_j \in G} m(\{f_j\}) f_j(a) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) \min\{f_i(a), f_j(a)\} \tag{6}$$

while monotonicity 1b) and normalization constraints 2b) become

1c) $m(\emptyset) = 0$, $\sum_{f_i \in G} m(\{f_i\}) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) = 1$,

2c) $\begin{cases} m(\{f_i\}) \geqslant 0, \text{ for all } f_i \in G, \\ m(\{f_i\}) + \sum_{f_j \in T} m(\{f_i, f_j\}) \geqslant 0, \text{ for all } f_i \in G \text{ and for all } T \subseteq G \setminus \{f_i\}, T \neq \emptyset. \end{cases}$

3.2. A motivating example

Let us show here the importance of taking into account the Choquet integral and its formulation in terms of 2-additive capacities.
Inspired by [38], let us suppose that the Dean of a school has to evaluate three students a, b, c whose scores on scientific (S) and humanistic (H) subjects are shown in Table 1. In particular, let us suppose

Table 1: Scores of three students on scientific and humanistic subjects

	$S(\cdot)$	$H(\cdot)$
a	30	23
b	23	30
c	25	25

that the Dean wants to formally represent their preferences for which they prefer c over the other two

students. Indeed, they prefer students who have balanced scores to students who are outstanding in a few subjects and above the minimum level in the others.
Trying to represent the preferences by a weighted sum (see eq. 2) where w_S is the weight of scientific subjects, w_H is the weight of humanistic subjects, and $w_S + w_H = 1$, one gets that

$$c \succ a \quad \Leftrightarrow \quad w_S \cdot S(c) + (1 - w_S) \cdot H(c) > w_S \cdot S(a) + (1 - w_S) \cdot H(a) \Leftrightarrow 25 > 7w_S + 23 \Leftrightarrow w_S < \frac{2}{7},$$

$$c \succ b \quad \Leftrightarrow \quad w_S \cdot S(c) + (1 - w_S) \cdot H(c) > w_S \cdot S(b) + (1 - w_S) \cdot H(b) \Leftrightarrow 25 > -7w_S + 30 \Leftrightarrow w_S > \frac{5}{7}.$$

Since the inequalities $w_S < \frac{2}{7}$ and $w_S > \frac{5}{7}$ are incompatible, the Dean's preferences cannot be represented by a weighted sum.
Trying to represent the same preferences by the Choquet integral expressed in terms of a capacity (see eq. 3) where $\mu(\{S\})$ is the weight of scientific subjects, $\mu(\{H\})$ is the weight of humanistic subjects and $\mu(\{S, H\}) = 1$ is the weight of scientific and humanistic subjects considered together, one gets that

$$c \succ a \quad \Leftrightarrow \quad 25 \cdot \mu(\{S, H\}) > 23 \cdot \mu(\{S, H\}) + (30 - 23) \cdot \mu(\{S\}) \Leftrightarrow \mu(\{S\}) < \frac{2}{7},$$

$$c \succ b \quad \Leftrightarrow \quad 25 \cdot \mu(\{S, H\}) > 23 \cdot \mu(\{S, H\}) + (30 - 23) \cdot \mu(\{H\}) \Leftrightarrow \mu(\{H\}) < \frac{2}{7}.$$

Considering, for example, the following capacity μ

$$\mu(\{S\}) = 0.25, \quad \mu(\{H\}) = 0.25, \quad \text{and} \quad \mu(\{S, H\}) = 1, \tag{7}$$

one is able to represent the Dean's preferences by the Choquet integral. Let us observe that since $\mu(\{S, H\}) > \mu(\{S\}) + \mu(\{H\})$, then, there is a positive interaction between scientific and humanistic subjects.
Considering the Möbius transformation of the capacity in (7), one has $\mu(\{S\}) = m(\{S\}) = 0.25$, $\mu(\{H\}) = m(\{H\}) = 0.25$ and $\mu(\{S, H\}) = m(\{S\}) + m(\{H\}) + m(\{S, H\})$ from which $m(\{S, H\}) = 0.5$. It is easy to observe that, in this case (and, in general, when only two criteria are considered), the computation of the Choquet integral in terms of a capacity (see eq. (3)) or in terms of Möbius (see eq. (5)) involves the same parameters number (3). However, computing the Choquet integral of one of the three alternatives in terms of Möbius, one gets

$$C_\mu(a) = C_\mu(S(a), H(a)) = \underbrace{m(\{S\}) \cdot S(a) + m(\{H\}) \cdot H(a)}_{WS(a)} + m(\{S, H\}) \cdot \min\{S(a), H(a)\}$$

clearly showing that the Choquet integral expressed in terms of Möbius is an extension of the weighted sum where the second part $(m(\{S, H\}) \cdot \min\{S(a), H(a)\})$ represents the eventual interactions between criteria.
Let us conclude this section by computing the Choquet integral of the alternatives a, b, c using the previous Möbius parameters ($m(\{S\}) = m(\{H\}) = 0.25$ and $m(\{S, H\}) = 0.5$) and eq. (6) obtaining

$$\begin{aligned} C_\mu(a) &= m(\{S\}) \cdot S(a) + m(\{H\}) \cdot H(a) + m(\{S, H\}) \cdot \min\{S(a), H(a)\} = \\ &= 0.25 \cdot 30 + 0.25 \cdot 23 + 0.5 \cdot 23 = 24.75, \\ C_\mu(b) &= m(\{S\}) \cdot S(b) + m(\{H\}) \cdot H(b) + m(\{S, H\}) \cdot \min\{S(b), H(b)\} = \\ &= 0.25 \cdot 23 + 0.25 \cdot 30 + 0.5 \cdot 23 = 24.75, \\ C_\mu(c) &= m(\{S\}) \cdot S(c) + m(\{H\}) \cdot H(c) + m(\{S, H\}) \cdot \min\{S(c), H(c)\} = \\ &= 0.25 \cdot 25 + 0.25 \cdot 25 + 0.5 \cdot 25 = 25 \end{aligned}$$

that perfectly represent the Dean's preferences.

3.3. NEMO-II-Ch

NEMO-II-Ch [10] is an interactive multiobjective optimization method aiming to drive the search for the most interesting Pareto front region for the DM. The method belongs to the family of NEMO[3] methods [9] which, based on NSGA-II, integrate some preferences provided by the DM during the iterations of the algorithm. The aim is to get points focused on a particular region of the Pareto front, avoiding wasting time surfing through regions not interesting for the DM. Initially, the model employs a straightforward weighted sum (2) as the preference function. However, if required, it switches to the 2-additive Choquet integral (6) when the weighted sum is no longer able to represent the DM's preferences.

Algorithm 1 NEMO-II-Ch method

```
1:  Current preference model = WEIGHTED SUM.
2:  Generate the initial population of solutions and evaluate them
3:  repeat
4:    if Time to ask the DM then
5:      Elicit user's preferences by asking DM to compare two randomly selected non-dominated
        solutions
6:      if there is no value function remaining compatible with the user's preferences then
7:        if Current preference model = WEIGHTED SUM then
8:          Current preference model = CHOQUET and go to 6:
9:        else
10:         Remove information on pairwise comparisons, starting from the oldest one, until fea-
            sibility is restored and reintroduce them in the reverse order as long as feasibility is
            maintained
11:       end if
12:     end if
13:     Rank solutions into fronts by iteratively identifying all solutions that are most preferred for
        at least one compatible value function. Rank within each front using crowding distance
14:   end if
15:   Select solutions for mating
16:   Generate offspring using crossover and mutation and add them to the population
17:   Rank solutions into fronts by iteratively identifying all solutions that are most preferred for at
      least one compatible value function. Rank within each front using crowding distance
18:   Reduce population size back to initial size by removing worst solutions
19: until Stopping criterion met
```

In the following, we shall describe the different steps in Algorithm 1:

1: As mentioned above, at the beginning, a weighted sum is used to represent the preferences of the DM;

2: An initial population of solutions is generated, and they are assessed based on the objective functions being considered;

4-5: If it is time to ask the DM for preference information, we order the solutions in fronts using the dominance relation, exactly as done in NSGA-II. The non-dominated solutions are put in

[3]NEMO: Necessary preference enhanced Evolutionary Multiobjective Optimizer

the first front. Once removed from the population, the other non-dominated solutions are put in the second front, and so on until all solutions have been ordered in different fronts. Inside the same front, the solutions are ordered using the crowding distance [23]. The DM is therefore presented with two non-dominated solutions. They are taken randomly from the first front (if there are at least two solutions) or from the following ones with at least two non-dominated solutions. In the extreme case in which there is only one solution for each front and, therefore, we have a complete order of the solutions, the DM is not presented with any pair of solutions, and we can pass to step 15:.
Let us suppose that solutions a and b have been chosen to be presented to the DM. They are asked to pairwise compare the two objective function vectors $(f_1(a), \ldots, f_n(a))$ and $(f_1(b), \ldots, f_n(b))$ stating if a is preferred to b ($a \succ_{DM} b$), b is preferred to a ($b \succ_{DM} a$) or a and b are indifferent ($a \sim_{DM} b$). A linear constraint will be used to translate this preference information. In particular, $a \succ_{DM} b$ is translated to the constraint $U(a) > U(b)$ and, $a \sim_{DM} b$ iff $U(a) = U(b)$. Let us observe that U is the function in (2) if the current preference model is the weighted sum, while U is the function in (6) if the current preference model is the 2-additive Choquet integral;

6: Checking if there exists at least one value function compatible with the preferences provided by the DM:

- If the current preference model is the weighted sum (2), then one has to solve the following LP problem:

$$\varepsilon_{DM}^{linear} = \max \varepsilon \text{ subject to}$$
$$\left.\begin{array}{l} U(a) \geqslant U(b) + \varepsilon, \text{ if } a \succ_{DM} b, \\ U(a) = U(b), \text{ if } a \sim_{DM} b, \\ \sum_{j=1}^{n} w_j = 1, \\ w_j \geqslant 0, \text{ for all } j = 1, \ldots, n. \end{array}\right\} E_{DM}^{linear}$$

  Let us observe that one constraint $U(a) \geqslant U(b) + \varepsilon$ should be included for all pairs $(a, b) \in A \times A$ for which the DM states that a is preferred to b ($a \succ_{DM} b$), while one constraint $U(a) = U(b)$ should be included for all pairs $(a, b) \in A \times A$ for which the DM states that a is indifferent to b ($a \sim_{DM} b$). If E_{DM}^{linear} is feasible and $\varepsilon_{DM}^{linear} > 0$, then there is at least one weighted sum compatible with the preferences provided by the DM.

- If the current preference model is the 2-additive Choquet integral in (6), then one has to solve the following problem:

$$\left.\begin{array}{l}
\varepsilon_{DM}^{Ch} = \max \varepsilon \text{ subject to} \\
C_\mu(w_1 f_1(a), \ldots, w_n f_n(a)) \geqslant C_\mu(w_1 f_1(b), \ldots, w_n f_n(b)) + \varepsilon, \text{ if } a \succ_{DM} b, \\
C_\mu(w_1 f_1(a), \ldots, w_n f_n(a)) = C_\mu(w_1 f_1(b), \ldots, w_n f_n(b)), \text{ if } a \sim_{DM} b, \\
w_j \geqslant 0, \text{ for all } j = 1, \ldots, n, \\
\displaystyle\sum_{j=1}^{n} w_j = 1, \\
m(\emptyset) = 0, \text{ and } \displaystyle\sum_{f_i \in G} m(\{f_i\}) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) = 1, \\
m(\{f_j\}) \geqslant 0, \text{ for all, } j = 1, \ldots, n, \\
m(\{f_j\}) + \displaystyle\sum_{f_i \in T} m(\{f_i, f_j\}) \geqslant 0, \text{ for all } j = 1, \ldots, n, \\
\text{and for all } T \subseteq \{f_1, \ldots, f_n\} \setminus \{f_j\}, T \neq \emptyset.
\end{array}\right\} E_{DM}^{Ch}$$

Let us underline that in the set of constraints above, we need to introduce a set of weights $(w_1, \ldots, w_n)$ so that $w_j \geqslant 0$ and $\sum_{j=1}^{n} w_j = 1$ since the Choquet integral application implies that all objectives are expressed on the same scale. Therefore, the set of weights is necessary to put the objectives on the same scale, and, for this reason, they become unknown variables of our model [10].

If E_{DM}^{Ch} is feasible and $\varepsilon_{DM}^{Ch} > 0$, then there is at least one value function, being a 2-additive Choquet integral, compatible with the preferences provided by the DM. Let us observe that the previous problem is no longer linear; consequently, we use the Nelder-Mead method [56] to get the set of weights and the Möbius parameters optimizing it. It is a numerical algorithm used to solve non-linear optimization problems that, iteratively, evaluates solutions belonging to a simplex. At each iteration, this simplex is transformed and the procedure continues until a stopping criterion is met (see [10] for a description of the application of the method in this context). The non-linearity of the problem comes from the constraints translating the preferences of the DM since, for all $a \in A$,

$$C_\mu(w_1 f_1(a), \ldots, w_n f_n(a)) = \sum_{j=1}^{n} w_j f_j(a) \cdot m(\{f_j\}) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) \cdot \min\{w_i f_i(a), w_j f_j(a)\}$$

and, consequently, $a \succ_{DM} b$ is translated into the constraint

$$\sum_{j=1}^{n} w_j f_j(a) \cdot m(\{f_j\}) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) \cdot \min\{w_i f_i(a), w_j f_j(a)\} \geqslant$$

$$\sum_{j=1}^{n} w_j f_j(b) \cdot m(\{f_j\}) + \sum_{\{f_i, f_j\} \subseteq G} m(\{f_i, f_j\}) \cdot \min\{w_i f_i(b), w_j f_j(b)\}.$$

Let us underline that in the above programming problems, the strict inequalities have been converted into weak inequalities by using an auxiliary variable ε, the maximization of which is the objective of our problems. For example, the strict inequality $U(a) > U(b)$ has been converted into the weak inequality $U(a) \geqslant U(b) + \varepsilon$;

7-10: If there is no model compatible with the preferences provided by the DM, we have to distinguish the case in which the current preference model is the weighted sum from the case in which the current preference model is the 2-additive Choquet integral. In the first case, given that no weighted sum is capable of representing the preferences of the DM, we opt to enhance the model's complexity by transitioning to the 2-additive Choquet integral. Having more degrees of freedom, it is more flexible and, therefore, can better adapt itself to the preferences of the DM. In the second case, if we already passed to the 2-additive Choquet integral but there is not any model (therefore weights and Möbius parameters) compatible with the preferences of the DM, we remove some pieces of this preference information starting from the oldest one until the feasibility is restored. Let us observe that removing a piece of preference information should be performed only if the DM agrees. This is a relevant aspect since the DM could be very convinced about a certain comparison and, consequently, they don't want to remove it;

13: To use the information gathered until now from the DM and, consequently, to address the search for the most interesting region of the Pareto front, we shall order the solutions in fronts in a different way than before. For each solution x in the current population (we shall denote by A the current set of solutions), we have to check if there is at least one compatible function such that x is strictly preferred to all other solutions in A. Again, we have to distinguish two cases:

- If the current preference model is the weighted sum one, the following LP problem has to be solved:

$$\varepsilon_x^{linear} = \max \varepsilon \ \text{ subject to,}$$
$$\left.\begin{array}{l} U(x) \geqslant U(a) + \varepsilon, \ \text{for all } a \in A \setminus \{x\}, \\ E_{DM}^{linear}. \end{array}\right\} E_x^{linear}$$

  If E_x^{linear} is feasible and $\varepsilon_x^{linear} > 0$, then x is put in the first front.

- If, instead, the current preference model is the 2-additive Choquet integral preference model, then the following programming problem has to be solved:

$$\varepsilon_x^{Ch} = \max \varepsilon \ \text{ subject to,}$$
$$\left.\begin{array}{l} C_\mu(w_1 f_1(x), \ldots, w_n f_n(x)) \geqslant C_\mu(w_1 f_1(a), \ldots, w_n f_n(a)) + \varepsilon, \ \text{for all } a \in A \setminus \{x\}, \\ E_{DM}^{Ch}. \end{array}\right\} E_x^{Ch}$$

  If E_x^{Ch} is feasible and $\varepsilon_x^{Ch} > 0$, then x is put in the first front.

Once the first front has been built, all solutions are removed from the current population, and the same procedure is used with the remaining solutions to build the second front. We shall continue in this way until all solutions have been ordered on different fronts. Inside the same front, solutions are ordered using the crowding distance.
In the rare case in which there is not any solution that can be preferred to the others for any compatible model, all solutions are retained equally preferable and, therefore, they are put in the same front;

15-18: The usual evolution of the population is performed by using the selection, crossover, and mutation operators together with the ordering of the population described above;

3-19: Repeat steps 4-18 until the stopping condition has been met.

4. Using Interactive Evolutionary Multiobjective Optimization in location problems: a case study

We test our approach on a well-known multiobjective location problem introduced in [27] and later in [28]. The problem, considered as a reference in its domain, consists in choosing the location of a given number p of facilities among a set of potential locations, optimizing five different classical objective functions for FLPs. More in detail, the facilities are Casualty Collection Points (CCPs) to which people can go if they need help in case disasters have happened. These centers should operate where a huge amount of people need to be provided with emergency services. In [27] a comparison of the different objectives is proposed and also a first multiobjective version, including only three objectives, is formulated; whereas in [28] a multiobjective heuristic has been introduced adopting the five objective functions described later. The problem is of particular interest among the MOFLPs because at least one $mini-max$ objective is selected, one for the $mini-sum$, and one equality measure are simultaneously optimized.

We define:

- $I = \{1, \ldots, q\}$: the set of demand points,
- $L = \{1, \ldots, m\}$: the set of potential locations for the facilities,
- d_{ij}: the distance between demand point i and potential facility j,
- pop_i: the population at the demand point i,
- p: the total number of facilities to locate,
- $P \subseteq L$: a vector of p selected facilities in L,
- $D_i(P)$: the distance from a demand point i to the closest facility in P,

$$D_i(P) = \min_{k \in P}\{d_{ik}\}.$$

We consider five objectives:

1. *The median objective*, minimizes the sum of the distances between the demand points and the closest facility [41]:

$$\min_P f_1(P) = \min_P \left[\frac{1}{q}\sum_{i=1}^{q} D_i(P)\right],$$

2. *The maximum distance objective*, minimizes the distance of the farthest demand point [41]:

$$\min_P f_2(P) = \min_P\{\max_i\{D_i(P)\}\},$$

3. *The maximum covering objectives*, maximize the population inside two different distance thresholds S_1 and S_2 [16]:

$$\max_P f_3(P) = \max_P \sum_{i:\, D_i(P)\leqslant S_1} pop_i,$$

$$\max_P f_4(P) = \max_P \sum_{i:\, D_i(P)\leqslant S_2} pop_i.$$

4. *The minimum variance objective*, balances the distances between demand points and the closest facility, minimizing the variance of the closest distances for all the demand points [53]:

$$\min_P f_5(P) = \min_P \frac{\sum_{i=1}^{q}[D_i(P) - f_1(P)]^2}{q}.$$

The case study has $I = \{1, \dots, 577\}$ demand points and $L = \{1, \dots, 141\}$ potential sites for the facilities located in Orange County in California, an area where the careful planning for the location of CCPs represents an essential requirement due to frequent earthquakes. The data, which include coordinates and associated weights for the demand points and coordinates for the potential facilities, are available upon request to the authors of [28].

Let us point out that this is just one of the possible examples that our methodology can handle. Our approach is very flexible, and we could adopt many different objective functions.

5. Algorithms used for the comparison

As already observed above, using a heuristic not taking into account the preferences of the DM, such as NSGA-II, gives the user a set of non-dominated vectors of p-facilities. The user then needs to select the most suitable solution based on their preferences. For this reason, we proposed to apply NEMO-II-Ch to address the search not to the entire Pareto front but to the most interesting part for the user.

We shall consider the full-size problem in which the 141 different locations will be taken into account, choosing the best p among them with $p = 4, 5$. Moreover, we will simulate different users' value functions. On the one hand, we will show that, in most of the cases, NEMO-II-Ch can find the best subset of p locations for the user by asking for a few pieces of preference information. On the other hand, to test its performance, we will compare them to the performance of three algorithms: EA-UVF, EA-UVF1, and EA-UVF2. These are based on the knowledge of the user's "true" value function that is, instead, unknown to the NEMO-II-Ch algorithm. While the EA-UVF algorithm has been presented in [10], its two variants, namely EA-UVF1 and EA-UVF2, are presented in this paper for the first time. The three algorithms are briefly described in the following sections.

5.1. EA-UVF: Evolutionary Algorithm based on User's Value Function

This algorithm has been presented in [10] and its main steps, which are listed in Algorithm 2 are detailed in the following lines:

Algorithm 2 Evolutionary Algorithm User's Value Function (EA-UVF) algorithm

1: Generate the initial population of solutions and evaluate them
2: Compute the utility of each solution by the user's true value function
3: Rank the solutions into fronts with respect to their true value
4: **repeat**
5:     Select solutions for mating
6:     Generate offspring using crossover and mutation and add them to the population
7:     Rank the solutions into fronts with respect to their true value
8:     Reduce population size back to initial size by removing worst solutions
9: **until** Stopping criterion met

1: Generate an initial population of solutions and evaluate them with respect to the considered objective functions;

2: Compute the utility of each solution by using the user's true value function;

3: Rank the solutions into fronts by using the values assigned to them from the user's true value function and computed at the previous step. The solution having the best utility value (the minimum [maximum] value if the user's true value function has to be minimized [maximized]) is put in the first front; the solution having the second best utility value is put in the second front and so on until the solution having the worst utility value that is included in the last front. Solutions having the same utility value are included in the same front;

5-8: Evolve the population;

4-9: Repeat steps 5-8 until the stopping condition has not been met.

5.2. EA-UVF1: NSGA-II with diversification replaced by User's Value Function

The steps of the EA-UVF1 algorithm are shown in Algorithm 3 and detailed in the following lines:

Algorithm 3 NSGA-II with diversification replaced by User's Value Function (EA-UVF1)

```
1: Generate the initial population of solutions and evaluate them
2: Rank solutions into fronts by dominance, and inside each front, order them using their true value

3: repeat
4:     Select solutions for mating
5:     Generate offspring using crossover and mutation and add them to the population
6:     Rank solutions into fronts by dominance, and inside each front, order them using their true
       value
7:     Reduce population size back to initial size by removing worst solutions
8: until Stopping criterion met
```

1: Generate an initial population of solutions and evaluate them with respect to the considered objective functions;

2: Rank solutions in non-dominated fronts. Then, inside each front, compute the true value of all solutions and rank them by these utility values;

4-7: Evolve the population;

3-8: Repeat steps 4-7 until the stopping condition has not been met.

The EA-UVF1 implements exactly the NSGA-II method with the replacement of the crowding distance, used to diversify solutions inside the same front, with the value assigned to the solutions by the user's true value function.

5.3. EA-UVF2: NSGA-II with a roulette wheel driven by User's Value Function

The steps of the EA-UVF2 algorithm are shown in Algorithm 4 and detailed in the following lines:

1: Generate an initial population of solutions and evaluate them with respect to the considered objective functions;

Algorithm 4 NSGA-II with a roulette wheel driven by User's Value Function (EA-UVF2)

```
1: Generate the initial population of solutions and evaluate them
2: repeat
3:     Assign a probability to be parent to each solution by using their true value
4:     Select solutions for mating
5:     Generate offspring using crossover and mutation and add them to the population
6:     Rank solutions into fronts by dominance, and inside each front, order them by the crowding
       distance
7:     Reduce population size back to initial size by removing worst solutions
8: until Stopping criterion met
```

3: A probability to be a parent of the next generation is assigned to each solution in the population. This probability, denoted by $Prob(P)$, is computed as

$$Prob(P) = \frac{U(P)}{\displaystyle\sum_{P \in POP} U(P)} \qquad \text{if } U \text{ has to be maximized,} \tag{8}$$

$$Prob(P) = \frac{\frac{1}{U(P)}}{\displaystyle\sum_{P \in POP} \frac{1}{U(P)}} \qquad \text{if } U \text{ has to be minimized} \tag{9}$$

and POP denotes the current population of solutions;

4-7: Evolve the population;

2-8: Repeat steps 3-7 until the stopping condition has not been met.

The EA-UVF2 algorithm follows all the steps of the NSGA-II method and assigns a probability for each solution to be a parent in the next generation based on the user's true value function. The better the value assigned by the user's true value function to a solution, the higher its probability of becoming a parent of the next generation.

Let us conclude this section by underlining that the EA-UVF represents the ideal situation where the algorithm has perfect knowledge of how the user decides between two solutions and thus has the greatest amount of theoretically available preference information. At the same time, the EA-UVF1 and the EA-UVF2 use this information, on the one hand, to select solutions within non-dominated fronts of the generated population and, on the other hand, to decide which solutions are the best to be parents of the next generation. However, all of them use the whole preference information that the DM could theoretically provide by preferentially ranking all solutions at all iterations of the evolutionary algorithm. In practice, its important to note that a DM cannot realistically provide all this preference information due to the excessive and impractical cognitive load resulting from making numerous comparisons during each iteration. Observe also that too much preference information could not be helpful for the optimization algorithm because it could prematurely steer to some uninteresting regions of the Pareto front. For these reasons, a methodology being much more parsimonious in asking preferences to the DM is requested for any real-world application. To investigate the most appropriate amount of preference information to request from the DM and obtain reasonably algorithmically acceptable solutions (i.e., to prevent the algorithm from being diverted to uninteresting areas of the Pareto front), we study the relationship between, on the one hand, the frequency with which preferences are requested from the user and, on the other hand, the quality of the results and the speed of convergence of the algorithms To this aim, in the following simulations, we run NEMO-II-Ch asking the DM one preference every 5, 10, and 20 generations, respectively.

6. Experimental setup and numerical results

The parameters and the technical details used in the simulations are the following:

- The population POP is composed of 30 solutions where each solution is a vector P of p different integer values taken in the interval $[1, m]$;
- The mating selection is performed by tournament selection in all methods apart from EA-UVF2 where it is performed by a roulette wheel selection:
  - *Tournament selection:* Let us denote by $P_1, \ldots, P_{30}$ the solutions in the current population. To each solution P_s is associated the front it belongs to (F_s). Moreover, in all methods each solution is associated with a *second score*. In NEMO-II-Ch and EA-UVF2 this second score is the crowding distance (CD_s)[4], while in EA-UVF1 the second score is the true value. We create a random permutation of the solutions in the population denoted by $P_{(1)}, \ldots, P_{(30)}$. Then, a tournament is performed between P_s and $P_{(s)}$ for each $s = 1, \ldots, 30$, to choose which solution has to be selected as parent of the next generation. The tournament is won from the solution being in the lowest front (P_s iff $F_s < F_{(s)}$ or $P_{(s)}$ iff $F_{(s)} < F_s$) or, if they belong to the same front ($F_s = F_{(s)}$), from the solution having the greatest second score. If P_s and $P_{(s)}$ belong to the same front, and they have the same second score, the winner is chosen randomly. Thus, thirty tournaments will be performed; consequently, 30 solutions will become parents of the next generation. Denoting by P'_s the winner of the tournament between P_s and $P_{(s)}$, the pairs of parents which will generate the offsprings of the next generation are, therefore, (P'_1, P'_2), (P'_3, P'_4),. . .,(P'_{29}, P'_{30});
  - *Roulette wheel selection:* Since, as in the tournament selection, 15 pairs of parents (P'_1, P'_2), (P'_3, P'_4),. . .,(P'_{29}, P'_{30}) have to be chosen, for each $k = 1, \ldots, 30$, a solution is sampled randomly from the probability distribution given by eq. (8) if the user's true value function U has to be maximized or by eq. (9) if the same function as instead to be minimized; the sampled solution becomes, therefore, the parent P'_k of the next generation;
- Each pair of parents generate two offsprings by one-point crossover with a probability of 1 and random resetting mutation[5] with a probability of $\frac{1}{p}$ [32]; in particular, since each solution can contain a certain location at most once, the one-point crossover has to be slightly modified if the two considered solutions have some common locations. In this case, the common potential location(s) are inherited by both offsprings, while the one-point crossover is performed on the two vectors composed of uncommon potential locations for both parents. For example, let us suppose that the two parents solutions are (10,15,21,30) and (6,10,20,50). In this case, the potential location labeled by 10 is present in both parents and, therefore, it is inherited by the two offsprings. The remaining vectors of uncommon locations are (15,21,30) and (6,20,50). The

[4]Citing [23], the crowding distance is ...*"the average distance of two points on either side of a particular solution along each of the objectives*, and it is computed to maintain the diversification of the population. The higher the crowding distance of a solution P_s, the more isolated the solution is in the considered population.

[5] *"...in each position independently, with probability p_m, a new value is chosen at random from the set of permissible values* [32]

one-point crossover is applied to exchange the two tails to these two vectors. Supposing that the cut point is the second integer, exchanging the two tails, we obtain the vectors (15,21,50) and (6,20,30). The two offsprings will therefore be the vectors (10,15,21,50) and (6,10,20,30). Let us underline that the evolution of the population is performed in such a way that if a new offspring is exactly the same as another solution in the current population, it is "killed. Therefore, it is not possible to have multiple copies of the same solutions in the population;

- Considering the set L of potential locations and a solution P composed of p of these potential locations, we assumed the following different user's value functions:

U^D) the maximal deviation from the optimal objective values [28] is computed as follows

$$U^D(P) = \max_{k\in\{1,\ldots,5\}} \{\Delta_k(P)\}$$

where

$$\Delta_k(P) = \begin{cases} \frac{f_k(P)-f_k^*}{f_k^*}, & \text{if the objective } f_k \text{ is to be minimized,} \\ \frac{f_k^*-f_k(P)}{f_k^*}, & \text{if the objective } f_k \text{ is to be maximized,} \end{cases}$$

and

$$f_k^* = \begin{cases} f_k^{min} = \min\limits_{\overline{P}\subseteq L:\, |\overline{P}|=p} f_k(\overline{P}), & \text{if the objective } f_k \text{ is to be minimized,} \\ f_k^{max} = \max\limits_{\overline{P}\subseteq L:\, |\overline{P}|=p} f_k(\overline{P}), & \text{if the objective } f_k \text{ is to be maximized,} \end{cases}$$

that is, f_k^* is the optimal value for the objective f_k, $k = 1, \ldots, 5$; a solution P is preferred to a solution P' if $U^D(P) < U^D(P')$;

U_v^D) On the basis of the U^D defined above, we considered the function U_v^D computed as follows:

$$U_v^D(P) = \max_{k\in v} \{\Delta_k(P)\}$$

where $v \in \{\{1,2,3,4\},\{1,2,3,5\},\{1,2,4,5\},\{1,3,4,5\},\{2,3,4,5\}\}$. In this way, we shall take into account only four of the five objective functions simultaneously;

U^N) the value is computed as follows

$$U^N(P) = \sum_{k=1}^{5} w_k \cdot \overline{f}_k(P)$$

where

$$\overline{f}_k(P) = \begin{cases} \frac{f_k(P)-f_k^{min}}{f_k^{max}-f_k^{min}}, & \text{if the objective } f_k \text{ is to be minimized,} \\ \frac{f_k^{max}-f_k(P)}{f_k^{max}-f_k^{min}}, & \text{if the objective } f_k \text{ is to be maximized,} \end{cases}$$

$w = (0.1, 0.15, 0.2, 0.25, 0.3)$, and a solution P is preferred to a solution P' if $U^N(P) < U^N(P')$;

U_v^N) the value is computed as follows

$$U_v^N(P) = \sum_{k\in v} w'_k \cdot \overline{f}_k(P)$$

where $w' = (0.1, 0.2, 0.3, 0.4)$ and $v \in \{\{1,2,3,4\},\{1,2,3,5\},\{1,2,4,5\},\{1,3,4,5\},\{2,3,4,5\}\}$[6]. Also in this case we consider a subset composed of four of the five objective functions and a solution P is preferred to a solution P' if $U_v^N(P) < U_v^N(P')$.

For all considered user's value functions, the best subset of p locations is $P_b \subseteq L$, such that $|P_b| = p$ and $U(P_b) = \min_{\overline{P} \subseteq L:\, |\overline{P}|=p} U(\overline{P})$ where $U \in \{U^D, U_v^D, U^N, U_v^N\}$;

- All algorithms are run for a maximum of 1,000 generations. In particular, for NEMO-II-Ch we asked the user to provide one preference comparison every 5, 10 and 20 generations. The resulting algorithms are therefore denoted by NIICh_5, NIICh_10 and NIICh_20. All the algorithms stop as soon as P_b is present in the current population or when the maximum number of generations has been reached.

After we described the setup of the simulations, let us present the results of applying the compared methods to the considered full-size problem. This means that we shall check for the best subset of p locations, with $p = 4, 5$, among the 141 taken into account. Of course, this problem is quite tricky since the possible subsets of p locations from which the best has to be discovered are $\binom{141}{4} = 15,777,195$ and $\binom{141}{5} = 432,295,143$, respectively. Therefore, we would like to prove that the method can deal with big-size problems in which a massive number of solutions is involved. We performed 50 independent runs for each of the twelve different users true value functions defined in the previous section (changing, therefore, the starting population), and we applied the three NEMO-II-Ch variants (NIICh_5, NIICh_10 and NIICh_20) as well as the three algorithms knowing the user's true value function (EA-UVF, EA-UVF1 and EA-UVF2).
In the tables below, we used the following performance measures and the corresponding notation to present the results of the simulations. Let us note that for U^N and U^D we related the performance measures to the 50 implemented runs for each of the users true value functions, while for the U_v^N and U_v^D we related the performance measures to the total number of runs implemented for each of the users true value functions for the five possible combinations of the four objectives, i.e., 250 for U_v^N considering $v \in \{\{1,2,3,4\},\{1,2,3,5\},\{1,2,4,5\},\{1,3,4,5\},\{2,3,4,5\}\}$ and further 250 for U_v^D considering $v \in \{\{1,2,3,4\},\{1,2,3,5\},\{1,2,4,5\},\{1,3,4,5\},\{2,3,4,5\}\}$:

- $\#SR$: number of runs in which the algorithm was able to discover the best subset P_b of possible locations;
- $M\#G$: mean number of generations necessary to the algorithm to discover P_b;
- $S\#G$: standard deviation of the number of generations necessary to the algorithm to discover P_b;
- $A\#P$: mean number of pairwise comparisons asked to the user in a single run and necessary to discover P_b. We did not include this data for EA-UVF and EA-UVF1 since they are only used as a benchmark and a comparison between the number of pairwise comparisons asked from the NEMO-II-Ch versions and the one involved in the application of both algorithms is meaningless. Of course, the number of times the user is queried by NEMO-II-Ch is only a small portion of the times the user has to provide a pairwise comparison in the two algorithms. Just to give an example, let us underline that in EA-UVF and EA-UVF1, where solutions are ranked with respect to the user's true value function, to rank order p solutions it is necessary to perform $\frac{p(p-1)}{2}$ pairwise comparisons[7]. This means that to rank order 30 solutions in the

[6] Let us observe that in the computation of $U_v^N(P)$ the functions $\overline{f}_k$ have a weight increasing with k. For example, if $v = \{1,3,4,5\}$, then, $U_v^N(P) = 0.1 \cdot \overline{f}_1(P) + 0.2 \cdot \overline{f}_3(P) + 0.3 \cdot \overline{f}_4(P) + 0.4 \cdot \overline{f}_5(P)$.

[7] The best solution is found after $p-1$ comparisons, the second after $p-2$ comparisons and so on.

population, the user has to provide 435 pairwise comparisons in a single iteration and, as will be clear in the next section, this number is much higher than the number of pairwise comparisons asked from the three NEMO-II-Ch versions in whichever considered test problem.
With respect to EA-UVF2, the user is not asked to provide any pairwise comparison. However, the algorithm can never be applied in practice since it is assumed that the user can assign a utility to each solution, a utility that needs to be used to implement the roulette wheel selection described above. Of course, this is not realistic at all;

- $S\#P$: standard deviation of the number of pairwise comparisons asked to the user necessary to discover P_b;
- MT: mean time (in seconds) necessary for the algorithm to discover P_b; all simulations have been performed using the commercial software MATLAB2019 but on different PCs. The 50 runs have been performed on the same machine for each method and user's value function. In the tables presenting the results, we reported the characteristics of the PCs used to perform the different simulations;
- ST: standard deviation of the time necessary to the algorithm to discover P_b[8];
- A_BRSD: the average distance of the best solution in the final population from the optimal solution P_b. The distance, denoted by $BRSD(U)$, is computed only for the simulations in which the algorithm was not able to discover P_b (in the case in which the algorithm can discover P_b the distance is zero). Denoting by P^{Best} the best solution in the final population, following [72], $BRSD(U)$ is computed as

$$BRSD(U) = \frac{|U(P^{Best}) - U(P_b)|}{U(P_b)}. \tag{10}$$

The less $BRSD(U)$, the better the algorithm's performance. The value A_BRSD is then obtained by averaging $BRSD(U)$ over the number of runs in which the algorithm could not discover P_b.

6.1. Comparison with EA-UVF, EA-UVF1 and EA-UVF2

In Tables 2-5 we reported the results of applying the three versions of NEMO-II-Ch and those obtained by the three algorithms knowing the user's true value function. We have considered the twelve different user's true value functions defined in the previous section and the cases $p = 4$ and $p = 5$ for the number of best locations to be discovered.
In the following, by (U, p) we denote the case in which the user's true value function is U, and the number of best locations is p. The following can be observed:

- $\left(U^N, 4\right)$ and $\left(U_v^N, 4\right)$:
  - *Convergence:* The three variants of NEMO-II-Ch as well as EA-UVF and EA-UVF1 are always able to find the best solution in the 50 runs. This is not the case for EA-UVF2 that, with respect to U^N is not able to converge in one of the 50 runs, while, with respect to U^N_{1245}, quite surprisingly, it can find the best subset of 4 locations only in 5 of the 50 runs;

[8] The mean and the standard deviation are computed for the runs in which P_b is discovered.

Table 2: Results for functions U^N and U_v^N considering $p = 4$.

U^N	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	**50/50**	**50/50**	**50/50**	**50/50**	**50/50**	**49/50**
$M\#G$	80.92	80.44	113.16	79.7	77.54	152.78
$S\#G$	55.28	46.99	76.45	50.34	49.16	137.43
$A\#P$	16.80	8.60	*6.16*			
$S\#P$	11.04	4.69	3.83			
MT	51.71s	36.40s	46.87s			
ST	43.99s	24.29s	33.16s			
A_BRSD						0.41
U_v^N	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	**250/250**	**250/250**	**250/250**	**250/250**	**250/250**	205/250
$M\#G$	98.38	104.30	129.47	74.81	92.18	159.53
$S\#G$	72.95	77.40	95.56	58.86	72.61	156.23
$A\#P$	20.31	11.00	6.99			
$S\#P$	14.56	7.72	4.82			
MT	111.36s	102.97s	89.63s			
ST	139.59s	113.46s	84.40s			
A_BRSD						0.233

Table 3: Results for functions U^N and U_v^N considering $p = 5$. All simulations have been performed with four different PCs which characteristics and labels are the following: (PC1) intel core i7 3.6GHz; (PC2) intel core i5 2.5GHz; (PC3) intel core i7 2.7GHz; (PC4) intel core i7 1.9GHz.

U^N	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	**50/50**	**50/50**	**50/50**	**50/50**	**50/50**	49/50
$M\#G$	140.94	156.08	184.96	112.22	137.20	269.08
$S\#G$	68.73	66.12	94.43	69.82	86.82	192.38
$A\#P$	28.82	16.16	*9.78*			
$S\#P$	13.75	6.63	4.68			
MT	135.11s	115.65s	118.8s			
ST	81.27s	63.21s	71.40s			
A_BRSD						0.127
U_v^N	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	**250/250**	249/250	247/250	245/250	**250/250**	189/250
$M\#G$	163.42	175.88	195.06	159.09	143.25	339.79
$S\#G$	141.67	129.87	131.35	151.05	110.22	257.24
$A\#P$	33.32	18.13	10.33			
$S\#P$	28.34	12.95	6.57			
MT	261.97s	225.75s	212.60s			
ST	406.12s	305.28s	267.40s			
A_BRSD		0.007	0.007	0.007		0.23

– *Convergence speed:* As can be observed from the data in Table 2, apart from the U^N case in which the EA-UVF1 converges more quickly (in terms of number of generations necessary to find P_b) than all the other algorithms, the EA-UVF is the quickest among the considered algorithms. As to the comparison between the three NEMO-II-Ch variants, on average, NIICh_5 converges more quickly than NIICh_10 in four of the six considered cases, while NIICh_20 is always the slowest. However, as already observed before, the

Table 4: Results for functions U^D and U_v^D considering $p = 4$.

U^D	NIICh_5	NIICh_10	NIICh_20	EA-UVF	EA-UVF1	EA-UVF2
$\#SR$	**50/50**	**50/50**	**50/50**	46/50	**50/50**	27/50
$M\#G$	223.16	216.64	211.72	275.72	221.40	340.52
$S\#G$	163.75	145.92	178.78	192.93	135.63	268.00
$A\#P$	45.30	22.22	11.14			
$S\#P$	32.76	14.62	8.91			
MT	6695.94s	3911.24s	1200.07s			
ST	12289.79s	8526.89s	3314.57s			
A_BRSD				0.217		0.091
U_v^D	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	**250/250**	**250/250**	**250/250**	207/250	239/250	172/250
$M\#G$	200.16	208.69	210.94	249.67	204.62	281.53
$S\#G$	156.24	163.38	183.76	185.98	144.05	244.34
$A\#P$	40.68	21.41	11.11			
$S\#P$	31.26	16.35	9.18			
MT	24304.79s	7808.71s	972.20s			
ST	74985.61s	18913.41s	3038.36s			
A_BRSD				0.13	0.08	0.09

number of pairwise comparisons asked by EA-UVF and EA-UVF1 is tremendously higher than the one involved in whichever NEMO-II-Ch version. For this reason, it is more meaningful to give a more in-depth analysis of the NEMO-II-Ch variants to understand if and how the number of times the user is queried with a pairwise comparison affects the convergence speed of the algorithm. It can be observed that the lowest number of pairwise comparisons is asked in correspondence of NIICh_20, followed by NIICh_10 and, then, by NIICh_5 (see values in italics). This means that not only NIICh_20 is efficient in finding P_b, but it can find it by asking very few pairwise comparisons to the user;

  – *Distance from P_b:* Considering EA-UVF2 and assuming that the best solution in the final population is the optimal one, the user makes an error, on average, of the 40.6% in the U^N case, and of the 23.3% in the U^N_{1245} one;

- $\left(U^N, 5\right)$ and $\left(U^N_v, 5\right)$:

  – *Convergence:* The three variants on NEMO-II-Ch can find P_b in all considered runs for all test problems apart from the case $\left(U^N_{1235}\right)$ in which NIICh_10 and NIICh_20 are not always able to find P_b. In particular, NIICh_10 does not find the best subset of five locations in one of the 50 runs, while NIICh_20 does not find the same subset of best locations in 3 out of the 50 runs.
  As to the three algorithms knowing the user's true value functions, EA-UVF1 can always find the best subset of five locations, while this is not true for the other two. In particular, EA-UVF does not find P_b in five of the fifty runs in the U^N_{1235} case, while EA-UVF2 has its best performance when U^N is considered (49/50), and its worst one in the case U^N_{1235}

Table 5: Results for functions U^D and U_v^D considering $p = 5$.

U^D	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	45/50	43/50	40/50	21/50	33/50	10/50
$M\#G$	433.33	365.49	455.20	214.71	508.92	337.70
$S\#G$	281.33	229.93	239.48	204.39	392.73	266.71
$A\#P$	87.31	37.19	23.25			
$S\#P$	56.25	22.95	12.00			
MT	49028.23s	17697.06s	6549.91s			
ST	45308.66s	26193.39s	8809.27s			
A_BRSD	0.038	0.093	0.104	0.103	0.101	0.175
U_{1234}^D	**NIICh_5**	**NIICh_10**	**NIICh_20**	**EA-UVF**	**EA-UVF1**	**EA-UVF2**
$\#SR$	193/250	185/250	178/250	114/250	144/250	36/250
$M\#G$	363.45	339.81	416.45	211.67	217.64	367.00
$S\#G$	259.70	227.57	236.80	175.07	161.94	289.25
$A\#P$	73.33	34.59	21.30			
$S\#P$	51.94	22.74	11.84			
MT	20765.65s	17287.27s	10285.38s			
ST	23766.28s	28641.42s	17749.90s			
A_BRSD	0.02	0.03	0.05	0.09	0.07	0.23

is the user's true value function (11/50). This suggests that using the user's true value function to assign a probability of becoming a parent of the next generation is worse than using the same function to rank the solutions belonging to the same front;

- *Convergence speed:* As in the $p = 4$ cases, it results that NIICh_20 is the quickest among the three NEMO-II-Ch versions to reach P_b since it asks the user to provide almost half of the pairwise comparisons asked by NIICh_10 and almost one-third of the pairwise comparisons asked by NIICh_5.
  Regarding EA-UVF and its two variants, once again, EA-UVF2 is the worst among them. Moreover, we would like to underline that the number of generations necessary to get P_b is lower for EA-UVF1 than for EA-UVF. In particular, it is meaningful to observe that the number of pairwise comparisons asked to the user by EA-UVF1 is not greater than the number of times the user is queried with a pairwise comparison in the EA-UVF. In fact, the application of the EA-UVF1 implies the same number of pairwise comparisons of EA-UVF only in case all solutions are non-dominated and, therefore, they are in one non-dominated front only. This suggests once again that parsimonious preference information is beneficial for the convergence of the algorithms to P_b;
- *Distance from P_b:* In the U_{1235}^N case, in average, the error done in assuming that the best solution in the last population is the optimal one is almost 7% for NIICh_10, NIICh_20 and EA-UVF, while it is 21.3% for the EA-UVF2. An higher error is also done by EA-UVF2 in the U_{1234}^N, U_{1245}^N and U_{2345}^N cases.

- $(U^D, 4)$ and $(U_v^D, 4)$:

- *Convergence:* The three variants of NEMO-II-Ch are always able to find P_b in all considered runs. This is not true for the three algorithms knowing the user's true value function. In particular, EA-UVF1 finds in all 50 runs the best subset of four locations for all user's true value functions apart from U^D_{1245} in which it finds P_b in 39 of the 50 runs; the EA-UVF never finds the best subset of locations in all runs. The same holds for EA-UVF2 that in the U^D_{1234} and U^D_{1245} cases finds P_b 49 and 47 times, respectively. Considering all other user's true value functions, it can find the best subset of four locations more or less half of the times;
- *Convergence speed:* NIICh_20 is confirmed as the best among the three variants of the NEMO-II-Ch since it asks a lower number of pairwise comparisons than the other two always maintaining the best possible convergence since, as observed in the previous item, it is always able to find P_b. Comparing NIICh_10 and NIICh_5, the first is better than the second in terms of number of pairwise comparisons asked to the DM;
- *Distance from P_b:* Assuming that the best solution in the last population is optimal, one makes an error ranging from 7.9% to 40.8% considering the EA-UVF, from 7.4% to 25.8% considering the EA-UVF2 and of the 7.8% considering the EA-UVF1.

- $\left(U^D, 5\right)$ and $\left(U^D_v, 5\right)$:
  - *Convergence:* For all considered cases, one of the three variants of NEMO-II-Ch finds P_b more often than the algorithms based on the knowledge of the user's true value function. Even more, in all cases the worst among the three NEMO-II-Ch variants performs at least as well as all three algorithms knowing the user's true value function in terms of number of runs in which it converges to P_b;
  - *Convergence speed:* Looking at the average number of pairwise comparisons asked to the user, once more we have the confirmation that NIICh_20 is the best among the three variants of NEMO-II-Ch since it finds P_b asking less pairwise comparisons than NIICh_5 and NIICh_10. However, differently from the previous cases, the doubt is now related to the fact that NIICh_20 is not able to find the best subset of locations as frequently as NIICh_5 and NIICh_10 and, therefore, it could be better to ask more pairwise comparisons to increase the probability to converge to the best solution.
  - *Distance from P_b:* Comparing the three versions of NEMO-II-Ch one can observe that, apart from U^D_{1235} and U^D_{1245} cases, NIICh_5 presents the best A_BRSD. In particular, the maximum average error is equal to 6.7% for NIICh_5, while it is 9.3% for NIICh_10 and even 10.4% for NIICh_20. The situation is even worse for the three algorithms based on the full knowledge of the user's true value function since, apart from the U^D_{1245} case in which the average error done assuming as an optimal solution the best solution in the final population is 1.9% considering EA-UVF1 and 4.6% considering EA-UVF, in all the other cases, this average error is at least equal to 9.8% with a pick of 51.8% done by EA-UVF2 in the U^D_{1234} case. This means that, in the case in which the EA-UVF algorithm and the other two variants cannot find P_b, they are very far from the area of the Pareto front most interesting with respect to the user's preferences.

To evaluate the significance of the data provided above we performed the Mann-Whitney U test with 5% significance level [45] to two different indicators:

1. considering $BRSD$ of each of the six algorithms in each of the 50 considered runs,
2. considering the number of pairwise comparisons asked to the user in each run for algorithms NIICh_5, NIICh_10 and NIICh_20.

Regarding the $BRSD$, we performed the test only for problems where at least one algorithm did not converge in at least one of the 50 runs. Indeed, if all methods had converged to the optimal solution in all runs, the $BRSD$ would be always equal to 0 and, consequently, the comparison between the algorithms would be absolutely meaningless.
Regarding the number of pairwise comparisons asked to the user, we performed the test on the NIICh_5, NIICh_10 and NIICh_20 only, since the number of pairwise comparisons asked to the user in EA-UVF, EA-UVF1 and EA-UVF2 is only virtual due to the unrealistic applicability of the algorithms. In particular, in the case in which the algorithm did not converge to the optimal solution, for that run, we considered the maximum number of pairwise comparisons asked to the user being 200 for NIICh_5, 100 for NIICh_10 and 50 for NIICh_20 since each of them asks one pairwise comparison every 5, 10 and 20 generations, respectively, and the maximum number of admitted generations is 1,000.

In the supplementary material, we included the results of the two tests. For brevity, we report here just the tables for the $(U^D, 5)$ case obtained performing the Mann-Whitney U test with 5% significance level on the $BRSD$ (Table 6) and on the number of pairwise comparisons asked to the user (Table 7). In both tables, we give the p-value together with the difference between the A_BRSD of each ordered pair of algorithms in Table 6 and the difference between $A\#P$ of each ordered pair of algorithms in Table 7. Bold values represent significant values considering the performed test.

Table 6: Mann-Whitney U test with 5% significance level performed on $BRSD$ for the $(U^D, 5)$. In the table, the p-value is provided, as well as the difference between the A_BRSD of each ordered pair of algorithms. In bold are the significant values.

U^D	NIICh_5	NIICh_10	NIICh_20	EA-UVF	EA-UVF1	EA-UVF2
NIICh_5		0.4598 (0.0038−0.013)	0.1137 (0.0038−0.0209)	**$5.28 \cdot 10^{-8}$** (0.0038−0.0595)	**0.0013** (0.0038−0.0345)	**$4.10 \cdot 10^{-13}$** (0.0038−0.1398)
NIICh_10			0.4355 (0.0130−0.0209)	**$9.59 \cdot 10^{-6}$** (0.013−0.0595)	**0.0220** (0.0130−0.0345)	**$3.31 \cdot 10^{-11}$** (0.013−0.1398)
NIICh_20				**0.0002** (0.0209−0.0595)	0.1517 (0.0209−0.0345)	**$6.42 \cdot 10^{-10}$** (0.0209−0.1398)
EA-UVF					**0.0120** (0.0595−0.0345)	**0.0001** (0.0595−0.1398)
EA-UVF1						**$1.36 \cdot 10^{-7}$** (0.0345−0.1398)

Table 7: Mann-Whitney U test with 5% significance level on the number of pairwise comparisons asked to the user in algorithms NIICh_5, NIICh_10 and NIICh_20 for the $(U^D, 5)$ case. In the table the p-value is provided as well as the difference between $A\#P$ of each ordered pair of algorithms. In bold the significant values.

U^D	NIICh_5	NIICh_10	NIICh_20
NIICh_5		**$1.82 \cdot 10^{-5}$** (98.68−46.12)	**$2.12 \cdot 10^{-9}$** (98.68−28.8)
NIICh_10			**0.0105** (46.12−28.8)

In Table 6 one can observe that the difference in the $BRSD$ between the NEMO variants is not significant, while the difference between the $BRSD$ of each NEMO variant and each of the algorithms based on the knowledge of the user's true value function is significant apart from the comparison between NIICh_20 and EA-UVF1 for which the difference between their $BRSD$ is not significant for the Mann-Whitney U test. This means that, on the one hand, the NEMO-II-Ch variants can be considered equivalent, while each of them is better than the three algorithms knowing the user's true value function. On the other hand, one can conclude that EA-UVF1 is better than EA-UVF which, in turn, is better than EA-UVF2.

Going at the data in Table 7, one can see that the difference between the distributions of the number of pairwise comparisons asked to the user in each pair of NEMO variants is significant. This means that the number of pairwise comparisons asked to the user by NIICh_20 to converge to the optimal solution is retained significantly smaller than the one involved in NIICh_10 and NIICh_5; consequently, with respect to the required preference information, NIICh_20 is better than NIICh_10 that, in turn, is better than NIICh_5.

Similar conclusions can be gathered by looking at all the other tables in the supplementary material. Once again, they confirm that the difference in the $BRSD$ between the NEMO variants and the algorithm based on the user's true value function is considered significant and that with respect to the three NEMO variants, the difference between the number of pairwise comparisons asked to the user from each algorithm is significant. The last fact proves that asking the user for less information does not affect, in general, the algorithmic capacity of NEMO-II-Ch to converge to the optimal solution.

7. Discussion

To prove the efficiency of the method in this setting, we considered a classical FLP very well-known in the literature [28] based on the most typical objective functions adopted in the domain. We performed different simulations running NEMO-II-Ch and comparing its performance with those of other three algorithms, namely EA-UVF, EA-UVF1, and EA-UVF2, based on the knowledge of the user's true value function that is, instead, unknown to NEMO-II-Ch.
In the comparison, we tested twelve different types of users' value functions and two different values for the number of facilities p that need to be located ($p = 4$ and $p = 5$). Moreover, to investigate how the number of comparisons asked to the user influences the convergence of the algorithm, we considered three different versions of the NEMO-II-Ch method, namely NIICh_5, NIICh_10, and NIICh_20, asking the user to compare one pair of non-dominated solutions every 5, 10 and 20 generations, respectively.

The results obtained should be read as an answer to the question: "*is there any methodological tool to handle real-world multiobjective facility problems*? The considered problem is very complex for the following reasons:

1. There is a plurality of objectives to be optimized,
2. Some of these objectives are quite complex in themselves (this is, in particular, the case of $f_5(P)$ [28]),
3. The preferences of the user have to be considered,
4. The preference information has to be collected while maintaining tolerable the cognitive burden for the DM,
5. The computation time should be acceptable for real-world operational applications.

Considering the number of pairwise comparisons requested by NEMO-II-Ch we have to conclude that it is acceptable. Indeed, it is interesting to compare our approach regarding the number of pairwise comparisons requested with one of the most well-known and most adopted MCDA methods, i.e., the AHP [67]. Let us consider the didactic example presented in [66] in which three schools (alternatives) are evaluated with respect to six different aspects (criteria). Regarding FLPs, it would be a really easy problem that will concern the selection of a single facility among three potential locations to optimize six different objectives. Since the DM must provide a pairwise comparison in terms of a qualitative judgment on a nine-point scale for each non-ordered pair of criteria and a comparison for each non-ordered pair of alternatives with respect to each criterion, the decision maker has to provide $\binom{6}{2} + 6\binom{3}{2} = 15 + 6 \cdot 3 = 33$ pairwise comparisons in total. This means that in a didactic example of, probably, the most adopted MCDA method [74], the DM is asked to give 33 pairwise comparisons. Looking again at the performance of NIICh_20, one can see that with a single exception, in all our cases, the algorithm was able to find the best solution with a number of pairwise comparisons much smaller than 33. Observe also that very often, the required average number of

pairwise comparisons asked to the user by NIICh_20 is lower than 15 (in 17 out of 24 considered cases). In addition, observe that while the pairwise comparisons of AHP require an evaluation on a nine-point scale, the pairwise comparisons considered in NEMO-II-Ch require simply to say which solution is preferred among the two. To have a more fair comparison between the judgments required by AHP and the information required by NEMO-II-Ch, consider that for each pair of items α and β being alternatives ($\alpha, \beta \in A$) or criteria ($\alpha, \beta \in G$) AHP requires, in fact, two comparisons: the first related to which one between α and β has the greatest priority and the second, expressed on the nine-point scale, related to how much greater is the priority of the item with the greatest priority with respect to the other. In general, it seems reasonable that the second comparison of AHP (the one on the nine-point scale) is more demanding than the pairwise comparison of NEMO-II-Ch related to which solution is the preferred among the two. Consequently, for each comparison asked by AHP on a pair of non-ordered items, one should assign a cognitive burden at least double with respect to the pairwise comparison required by NEMO-II-Ch. Let us note that we used an even pattern of interactions, maintaining a constant number of generations between each interaction, and we did not explore the impact of changing to other patterns such as front-loaded or rear-loaded [52]. It is surely something that could be explored as an avenue for future research. Additionally, we did not consider DM inconsistencies and we assumed that the pairwise comparisons were correctly performed. In conclusion, we can say that, on average, NEMO-II-Ch can handle a quite challenging problem with a complexity comparable to that of one of the most demanding real-world problems, asking the user a cognitive burden much smaller than the one required by the most adopted MCDA method in a very didactic example.
Coming to the computational time, even considering the case taking more time, NIICh_20 is almost always (apart from one case only) achieving the optimal solution, on average, in less than three hours and, very often, in less than one hour (quite frequently in the U^N and U_v^N cases in some minutes). This seems a very reasonable running time for such a complex problem.

Beyond the specific interest in the multiobjective facility location problems, the results we obtained are also relevant from the general point of view of the multiobjective optimization algorithms. The procedure that has been proposed can be seen as a parsimonious exploration of the space of solutions and the DM's preferences. The parsimony of the multiobjective optimization procedure we have applied can be decomposed into two components:

- a component related to the optimization procedure: it is based on the evaluations of combinations of the most promising solutions maintaining a certain level of diversification typical of the evolutionary algorithms,
- a component related to the preference learning procedure: it is based on a "dynamical induction of the DM's utility function" based on few preference comparisons, typical of the ordinal regression approach [46] that is properly applied in an "incremental version" adding time by time preferences related to new solutions discovered by the optimization algorithm.

Note that while EA-UVF1, functioning as NSGA-II but with diversification changed to the User's value function, can enhance EA-UVF's performance, it still falls short of achieving the same efficiency as NIICh_20. Taking into consideration the number of runs in which the optimal solution was discovered, NIICh_20 can obtain better results than EA-UVF1 in 6 cases, while EA-UVF1 can perform better than NIICh_20 in one case only. We believe that this can be interpreted in the sense that the parsimony in the required preference information of NIICh_20 permits us to obtain better performance of an algorithm using the whole preference information as EA-UVF1. To sum up, our findings from the multiobjective facility location problem indicate that for highly intricate combinatorial optimization problems, employing an evolutionary algorithm and a minimal elicitation of the DM's preference information can be a suitable strategy. Of course, this hypothesis needs to be tested on other multiobjective combinatorial problems and, more generally, on other complex multiobjective problems (not necessarily combinatorial), to obtain a more precise and definitive confirmation.

8. Conclusions

We considered a very complex problem resulting from combining two other complex problems. The combination of the two problems highly exacerbates the difficulty. The two problems are the facility location problem and the search for optimal solutions in multiobjective decision problems taking into account the user's preferences. In this perspective, the research question of the paper is: "*Is it possible to give an adequate answer, especially taking into account real-world applications, to the so complex problem resulting from the combination of the above-mentioned problems?*" Technically, the answer to the problem is obtained from applying a state-of-the-art multiobjective optimization procedure to the standard formulation of a multiobjective facility location problem. The contribution of the paper is in handling the question and in providing a surprisingly very positive answer: the two complex problems can be solved together with a reasonable cognitive burden (comparable and even smaller than the cognitive burden required from didactic examples of the most adopted MCDA methods) and with reasonable computational times (especially considering the use of non-specialized programming languages and the computation on common laptops daily used). Apart from applying the presented methodology to other complex multiobjective combinatorial optimization problems to gather additional evidence on its effectiveness and reliability in such intricate decision-making scenarios, the following potential research directions can be emphasized:

- Research should be addressed on determining the optimal frequency at which users should be prompted to provide preference information, in order to expedite the convergence of the algorithm. Additionally, investigating techniques that determine which solutions should be presented to the user in order to enhance the algorithm's learning capabilities can also contribute to improving convergence [11, 17];
- In order to address larger real-world problems, a more efficient implementation of NEMO-II-Ch needs to be developed. Upon examining the computational time required to run the algorithm, it becomes clear that nearly 93% of the time is consumed by the execution of the Nelder-Mead method. Integrating alternative methods for solving non-linear optimization problems could significantly accelerate the algorithm and enhance its practical applicability;
- Considering the favorable outcomes achieved with NEMO-II-Ch in addressing location problems, we believe it would be worthwhile to explore its application in various other classical combinatorial optimization problems that can formulated from a multiobjective standpoint such as the ones described in [25] and [42].

Acknowledgements

The authors are grateful to Professor Tammy Drezner for making available the data concerning the real-world problem.

Funding

The second and the third authors wish to acknowledge the support of the Ministero dell'Istruzione, dell'Universitá e della Ricerca (MIUR) - PRIN 2017, project Multiple Criteria Decision Analysis and Multiple Criteria Decision Theory, grant 2017CY2NCA.

References

[1] L. Alcada-Almeida, J. Coutinho-Rodrigues, and J. Current. A multiobjectrive modeling approach to Locating incinerators. *Socio-Economic Planning Sciences*, 43(2):111–120, 2009.

[2] M.J. Alves and J. Clímaco. A review of interactive methods for multiobjective integer and mixed-integer programming. *European Journal of Operational Research*, 180(1):99–115, 2007.

[3] S. Angilella, S. Greco, and B. Matarazzo. Non-additive robust ordinal regression: A multiple criteria decision model based on the Choquet integral. *European Journal of Operational Research*, 201(1):277–288, 2010.

[4] M. Basirati, R. Billot, and P. Meyer. Two parameter-tuned multi-objective evolutionary-based algorithms for zoning management in marine spatial planning. *Annals of Mathematics and Artificial Intelligence*, pages 1–32, 2023.

[5] O. Berman, Z. Drezner, and D. Krass. Generalized coverage: new developments in covering location models. *Computer & Operations Research*, 37(10):1675–1687, 2010.

[6] R. Bhattacharya and S. Bandyopadhyay. Solving conflicting bi-objective facility location problem by NSGA II evolutionary algorithm. *The International Journal of Advanced Manufacturing Technology*, 51(1-4):397–414, 2010.

[7] R. Blanquero and E. Carrizosa. A DC biobjective location model. *Journal of Global Optimization*, 23(2):139–154, 2002.

[8] J. Branke, K. Deb, K. Miettinen, and R. Słowiński, editors. *Multiobjective Optimization: Interactive and Evolutionary Approaches*, volume 5252 of *LNCS*. Springer, Berlin, 2008.

[9] J. Branke, S. Greco, R. Słowiński, and P. Zielniewicz. Learning Value Functions in Interactive Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 19(1):88–102, 2015.

[10] J. Branke, S. Corrente, S. Greco, R. Słowiński, and P. Zielniewicz. Using Choquet integral as preference model in interactive evolutionary multiobjective optimization. *European Journal of Operational Research*, 250:884–901, 2016.

[11] J. Branke, S. Corrente, S. Greco, and W.J. Gutjahr. Efficient pairwise preference elicitation allowing for indifference. *Computers and Operations Research*, 88:175–186, 2017.

[12] H. Calik, M. Labbé, and H. Yaman. p-Center problems. In *Location Science*, pages 79–92. Springer, 2015.

[13] E. Carrizosa, A. Ushakov, and I. Vasilyev. Threshold robustness in discrete facility location problems: a bi-objective approach. *Optimization Letters*, 9(7):1297–1314, 2015.

[14] A. Charnes and W. W. Cooper. Goal programming and multiple objective optimizations: Part 1. *European Journal of Operational Research*, 1(1):39–54, 1977.

[15] G. Choquet. Theory of capacities. *Annales de l'Institut Fourier*, 5(54):131–295, 1953.

[16] R.L. Church and C.S. ReVelle. The maximal covering location problem. *Papers in Regional Science*, 32(1):101–118, 1974.

[17] K. Ciomek, M. Kadziński, and T. Tervonen. Heuristics for prioritizing pair-wise elicitation questions with additive multi-attribute value models. *Omega*, 71:27–45, 2017.

[18] C.A. Coello Coello. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, 191(11-12):1245–1287, 2002.

[19] J. Coutinho-Rodrigues, L. Tralhão, and L. Alçada-Almeida. A bi-objective modeling approach applied to an urban semi-desirable facility location problem. *European Journal of Operational Research*, 223(1):203–213, 2012.

[20] P. Czyżżak and A. Jaszkiewicz. Pareto simulated annealinga metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-criteria Decision Analysis*, 7 (1):34–47, 1998.

[21] M.S. Daskin. *Network and discrete location: models, algorithms, and applications*. Wiley, New York, USA, 1995.

[22] K. Deb. *Multi-objective optimization using evolutionary algorithms*. Chichester, UK: Wiley, 2001.

[23] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[24] J. Dias, M.E. Captivo, and J. Clímaco. A memetic algorithm for multi-objective dynamic location problems. *Journal of Global Optimization*, 42(2):221–253, 2008.

[25] K.F. Doerner, W.J. Gutjahr, and P.C. Nolz. Multi-criteria location planning for public facilities in tsunami-prone coastal areas. *Or Spectrum*, 31(3):651–678, 2009.

[26] P. Domínguez-Marín. *The Discrete Ordered Median Problem: Models and Solution Methods*. Springer Science & Business Media, 2013.

[27] T. Drezner. Location of casualty collection points. *Environment and Planning C: Government and Policy*, 22(6):899–912, 2004.

[28] T. Drezner, Z. Drezner, and S. Salhi. A multi-objective heuristic approach for the casualty collection points location problem. *Journal of the Operational Research Society*, 57(6):727–734, 2006.

[29] Z. Drezner and H.M. Hamacher. *Facility location: applications and theory*. Springer, New York, USA, 2001.

[30] M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR Spectrum*, 22(4):425–460, 2000.

[31] M. Ehrgott and X. Gandibleux. Hybrid Metaheuristics for Multi-objective Combinatorial Optimization. *Studies in Computational Intelligence*, 114:221–259, 2008.

[32] A.E. Eiben and J.E. Smith. *Introduction to Evolutionary Computing*. Springer, 2003.

[33] H.A. Eiselt and G. Laporte. *Objectives in location problems*. Springer-Verlag, New York, USA, 1995.

[34] R. Z. Farahani, M. SteadieSeifi, and N. Asgari. Multiple criteria facility location problems: A survey. *Operations Research*, 34(7):1689–1709, 2010.

[35] H.A. Fernandes and V. Marianov. *Foundations of location analysis*. International Series in Operations Research and Management Science. Springer, New York, USA, 2011.

[36] S. Fernandes, M.E. Captivo, and J. Clímaco. A DSS for bicriteria location problems. *Decision Support Systems*, 57:224–244, 2014.

[37] J. W. Fowler, E. S. Gel, M. Köksalan, P. Korhonen, J. L. Marquis, and J. Wallenius. Interactive evolutionary multi-objective optimization for quasi-concave preference functions. *European Journal of Operational Research*, 206(2):417–425, 2010.

[38] M. Grabisch. The application of fuzzy integrals in multicriteria decision making. *European Journal of Operational Research*, 89(3):445–456, 1996.

[39] M. Grabisch and C. Labreuche. Fuzzy measures and integrals in MCDA. In S. Greco, M. Ehrgott, and J.R. Figueira, editors, *Multiple Criteria Decision Analysis: State of the Art Surveys*, pages 553–603. Springer, New York, NY, 2016.

[40] S. Greco, M. Ehrgott, and J.R. Figueira. *Multiple Criteria Decision Analysis: State of the Art Surveys*. Springer, New York, 2016.

[41] S.L. Hakimi. Optimum location of switching center and the absolute centers and medians of a graph. *Operations Research*, 12(3):450–459, 1964.

[42] H.W. Hamacher, M. Labbe, S. Nickel, and A.J. Skriver. Multicriteria semi-obnoxious network location problems (MSNLP) with sum and center objectives. *Annals of Operations Research*, 110(1-4):33–53, 2002.

[43] I. Harris, C.L. Mumford, and M.M. Naim. An evolutionary bi-objective approach to the capacitated facility location problem with cost and CO2 emissions. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pages 697–704. ACM, 2011.

[44] A.M. Heyns and J.H. van Vuuren. Multi-objective optimisation of discrete GIS-based facility location problems. *Optimization and Engineering*, 2015.

[45] M. Hollander, D.A. Wolfe, and E. Chicken. *Nonparametric statistical methods*, volume 751. John Wiley & Sons, 2013.

[46] E. Jacquet-Lagreze and Y. Siskos. Assessing a set of additive utility functions for multicriteria decision-making, the UTA method. *European Journal of Operational Research*, 10(2):151–164, 1982.

[47] J. Kalcsics, S. Nickel, M.A. Pozo, J. Puerto, and A.M. Rodríguez-Chía. The multicriteria p-facility median location problem on networks. *European Journal of Operational Research*, 235 (3):484–493, 2014.

[48] E. Karasakal and D. Nadirler. An interactive solution approach for a bi-objective semi-desirable location problem. *Journal of Global Optimization*, 42(2):177–199, 2008.

[49] R.L. Keeney and H. Raiffa. *Decisions with multiple objectives: Preferences and value tradeoffs*. J. Wiley, New York, 1976.

[50] J. Krarup and P.M. Pruzan. The simple plant location problem: survey and synthesis. *European Journal of Operational Research*, 12(1):36–81, 1983.

[51] G. Laporte, S. Nickel, and F.S. da Gama. *Location science*. Springer, Berlin, 2015.

[52] J. Marquis, E. S. Gel, J. W. Fowler, M. Köksalan, P. Korhonen, and J. Wallenius. Impact of number of interactions, different interaction patterns, and human inconsistencies on some hybrid evolutionary multiobjective optimization algorithms. *Decision Sciences*, 46(5):981–1006, 2015.

[53] M.T. Marsh and D.A. Schilling. Equity measurement in facility location analysis: A review and framework. *European Journal of Operational Research*, 74(1):1–7, 1994.

[54] K. Miettinen, F. Ruiz, and A.P. Wierzbicki. Introduction to multiobjective optimization: interactive approaches. In J. Branke, K. Deb, R. Słowiński, and K. Miettinen, editors, *Multiobjective optimization*, pages 27–57. Berlin: Springer, 2008.

[55] N. Mladenovic, J. Brimberg, P. Hansen, and J.A. Moreno-Perez. The p-median problem: A survey of metaheuristic approaches. *European Journal of Operational Research*, 179(3):927–939, 2007.

[56] J.A. Nelder and R. Mead. A simplex method for function minimization. *The computer journal*, 7(4):308–313, 1965.

[57] P. Nijkamp and J. Spronk. Interactive multidimensional programming models for locational decisions. *European Journal of Operational Research*, 6(2):220–223, 1981.

[58] Y. Ohsawa and K. Tamura. Efficient location for a semi-obnoxious facility. *Annals of Operations Research*, 123(1-4):173–188, 2003.

[59] Y. Ohsawa, N. Ozaki, and F. Plastria. Equity-effciency bicriteria location with squared Euclidean distances. *Operations Research*, 56(1):79–87, 2008.

[60] S.H. Owen and M.S. Daskin. Strategic facility location: a review. *European Journal of Operational Research*, 111(3):423–447, 1998.

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

[62] S.H.A. Rahmati, A. Ahmadi, M. Sharifi, and A. Chambari. A multi-objective model for facility location–allocation problem with immobile servers within queuing framework. *Computers & Industrial Engineering*, 74:1–10, 2014.

[63] J. Rakas, D. Teodorović, and T. Kim. Multi-objective modeling for determining location of undesirable facilities. *Transportation Research Part D: Transport and Environment*, 9(2):125–138, 2004.

[64] B. Roy. Meaning and validity of interactive procedures as tools for decision making. *European Journal of Operational Research*, 31(3):297–303, 1987.

[65] B. Roy. Decision science or decision-aid science? *European Journal of Operational Research*, 66 (2):184–203, 1993.

[66] T. Saaty. A scaling method for priorities in hierarchical structures. *Journal of Mathematical Psychology*, 15(3):234–281, 1977.

[67] T. Saaty. *The Analytic Hierarchy Process*. New York, McGraw-Hill, 1980.

[68] T. Schnepper, K. Klamroth, M. Stiglmayr, and J. Puerto. Exact algorithms for handling outliers in center location problems on networks using k-max functions. *European Journal of Operational Research*, 273(2):441–451, 2019.

[69] P. Serafini. Some considerations about computational complexity for multi objective combinatorial problems. In J. Jahn and W. Krabs, editors, *Recent Advances and Historical Development of Vector Optimization*, pages 222–232. Springer, 1987.

[70] B.L. Shankar, S. Basavarajappa, J.C.H. Chen, and R.S. Kadadevaramath. Location and allocation decisions for multi-echelon supply chain network–A multi-objective evolutionary approach. *Expert Systems with Applications*, 40(2):551–562, 2013.

[71] B.C. Tansel, R.L. Francis, and T.J. Lowe. A biobjective multifacility minimax location problem on a tree network. *Transportation Science*, 16(4):407–429, 1982.

[72] M.K. Tomczyk and M. Kadzinski. EMOSOR: Evolutionary multiple objective optimization guided by interactive stochastic ordinal regression. *Computers & Operations Research*, 108:134 – 154, 2019.

[73] M.K. Tomczyk and M. Kadzinski. Decomposition-based co-evolutionary algorithm for interactive multiple objective optimization. *Information Sciences*, 549:178 – 199, 2021.

[74] O.S. Vaidya and S.l Kumar. Analytic hierarchy process: An overview of applications. *European Journal of operational research*, 169(1):1–29, 2006.

[75] J.G. Villegas, F. Palacios, and A.L. Medaglia. Solution methods for the bi-objective (cost-coverage) unconstrained facility location problem with an illustrative example. *Annals of Operations Research*, 147(1):109–141, 2006.

[76] H. Yapicioglu, A.E. Smith, and G. Dozier. Solving the semi-desirable facility location problem using bi-objective particle swarm. *European Journal of Operational Research*, 177(2):733–749, 2007.

[77] A. Zhou, B.Y. Qu, H. Li, S.Z. Zhao, P.N. Suganthan, and Q. Zhang. Multiobjective evolutionary algorithms: A survey of the state of the art. *Swarm and Evolutionary Computation*, 1(1):32–49, 2011.

[78] S. Zionts and J. Wallenius. An interactive programming method for solving the multiple criteria problem. *Management Science*, 22(6):652–663, 1976.

[79] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization. In K.C. Giannakoglou et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE), 2002.

Highli

- We deal with Multiobjective Combinatorial Optimization (MOCO) problems
- We use Interactive Evolutionary Multiobjective Optimization (IEMO) methods for MOCO problems
- IEMO integrates preferences provided by Decision Makers in the search procedure
- We applied a IEMO methodology, called NEMO-II-Ch, to facility location problems

onflict of I

Declaration of interests

☒ The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

☐The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Journal Pre-proof