$R2$-IBMOLS applied to a practical case of the multiobjective knapsack problem

Brahim Chabane[a,b], Matthieu Basseur[a], Jin-Kao Hao[a,c,*]

[a] *LERIA, Université d'Angers, 2 Bd Lavoisier, 49045 Angers, France*
[b] *GePI Conseil, Grand Maine - Allée du Grand Launay, 49000 Angers, France*
[c] *Institut Universitaire de France, Paris, France*
Expert Systems With Applications, DOI: 10.1016/j.eswa.2016.11.007

Abstract

The social and medico-social sector is experiencing a fast evolution due to the continuing growth of older population. Yet, social and medico-social structures suffer from a real lack of computerized decision support tools. This work deals with the key issue of elaborating efficient action plans in these structures, which aims to improve the whole quality of these structures. An efficient action plan is a set of actions chosen among many candidate actions which optimize several conflicting objectives and satisfy some imperative constraints. To assist managers to optimize their action plans, we develop a multiobjective decision support system as part of a commercial software. According to the objectives and constraints defined by the decision maker and a set of feasible actions, the software is used to select the actions that optimize the given objectives while satisfying the constraints. After providing a description and a formal model of the action plan optimization problem, we present a solution method using the iterated local search based on quality indicators (IBMOLS). We assess the proposed approach on problem instances with 2 to 8 objectives and up to 500 candidate actions and demonstrate its usefulness as a key component of a decision support system for social and medico-social structures.

Keywords: Action planning; Multiobjective optimization; Decision support; Heuristics.

1. Introduction

The problem studied in this paper concerns the social and medico-social sector in France, which includes more than 34,000 different structures (rest houses, accommodation and rehabilitation centers, work-based support centers,

*Corresponding author.
Email addresses: chabane@info.univ-angers.fr (Brahim Chabane), basseur@info.univ-angers.fr (Matthieu Basseur), hao@info.univ-angers.fr (Jin-Kao Hao)

etc) (French SAH Ministry, 2012). The social and medico-social sector is in perpetual and fast evolution, especially since the French law No. 2002-2 renovating social and medico-social actions, which considers the actions covering planning, programming, resource allocation, structure evaluation and coordination as a fundamental basis for the management of *social and medico-social structures* (structures for short hereafter). In this context, elaborating efficient action plans becomes an important and challenging task for decision makers in these structures.

In this work, we investigate a real case of multiobjective action plan optimization problem. This study is unique because even if the social and medico-social sector is increasingly computerized these last years, it remains among the sectors where optimization is not yet used as a tool for decision support. Indeed, the use of computer resources has made considerable progress in this sector, yet they are basically employed for the daily management of the structures and optimization tools are usually completely absent. In this context of lack of advanced models and tools, we develop a multiobjective decision support system to assist managers to optimize their action plans, which, to our knowledge, is the first study of this kind for the social and medico-social sector. This study illustrates the interest of multiobjective optimization for building expert and intelligent systems for decision making.

The problem consists in elaborating optimized action plans in order to improve the overall management efficiency of structures. The aim is to choose a subset of actions among many possible actions while optimizing several objectives and satisfying some imperative constraints (e.g., budget). Each action has a realization cost and can influence, positively or negatively, some or all the objectives. The global cost of the final solution (i.e., an action plan) should not exceed a predefined budget. Also, a threshold constraint could be added to each objective indicating the minimal objective value that solutions must attain. Thus, this problem can be considered as a practical case of the multiobjective knapsack problem (MOKP) (Barichard and Hao, 2003; Lust and Teghem, 2012), where actions represent items (objects) which need to be selected to put in a knapsack whose capacity is defined by the budget allocated to the project while optimizing a number of objectives. We note that, in the real case, we may need to consider more than one thousand possible actions and up to eight objectives.

In this paper, we present an application of the Indicator-based Multiobjective Local Search (IBMOLS) algorithm (Basseur and Burke, 2007; Basseur et al., 2012) with the Epsilon (ϵ) indicator (Zitzler and Künzli, 2004) and the $R2$ indicator (Hansen and Jaszkiewicz, 1998; Brockhoff et al., 2012) to the action plan optimization problem. We first present a formal model of the action plan optimization. This modeling allows us to associate the action plan optimization problem to the well-known multiobjective knapsack problem. Then, we show that IBMOLS coupled with the $R2$ indicator is efficient even on large size problem instances. This is the first time that IBMOLS is used with the $R2$ indicator to solve a multiobjective optimization problem with more than three objectives. This work was partially motivated by a preliminary study using the Epsilon (ϵ) indicator (Chabane et al., 2015), which showed that this indicator

does not always maintain naturally population diversity in the objective space. In this work, we show that $R2$-IBMOLS is more efficient than ϵ-IBMOLS and NSGA-II (Deb et al., 2002) on all the instances tested and can help decision makers to elaborate action plans which improve the overall quality of their structures. More generally, this study demonstrates that multiobjective optimization constitutes a highly useful tool to build expert and intelligent decision support systems for the fast evolving social and medico-social sector.

The paper is organized as follows. In Section 2, we present the action plan optimization problem and its formulation as a multiobjective knapsack problem. In Section 3, we provide basic definitions about multiobjective optimization and the binary indicator search principle. In Section 4, we describe the IBMOLS algorithm for action plan optimization. In Section 5, we present the problem instances and the experimental protocol for computational assessments. In Section 6, we show our experimental results and comparative studies, followed by a discussion on the practical impact on action plan elaboration in Section 7. In Section 8, we provide concluding comments and perspectives.

2. Action plan optimization problem

This section describes the context and interest of optimized action plan elaboration to improve the whole quality of social and medico-social structures. In the second part, we present a mathematical formulation of the action plan optimization problem after recalling the general definition of the MOKP.

2.1. Problem description

The action plan optimization problem is a practical case of the MOKP, which is well known in the literature. The MOKP often appears in the field of project management as well as several other domains. This work is a part of the "*MSQualité*"software developed by the company *GePI* [1] which works specifically for the social and medico-social sector in France. Since the law No. 2002-2 of January 2*nd*, 2002, social and medico-social structures are constrained to continuously elaborate improvement projects, carry out at least one self-assessment every 5 years and one external assessment (assessment done by a person outside the structure) every 7 years. This assessment aims to evaluate the operating structure, its improvement project and the quality of service offered to the persons entering in the structure. At the end of the evaluation process, recommendations are given to the decision maker to conduct a new project for the next period. To elaborate the project, the decision maker defines the objectives to achieve, the resources to use, the constraints and the actions plan to implement. Some precisions and examples about the objectives, actions and constraints typically found in projects are given below.

[1] http://www.gepi-conseil.com

- *Objectives*

  The objectives depend mainly on the recommendations resulting from the last assessment (self-assessment or external assessment). They could be of varied nature, either concerning the residents lives such as *"improve resident's life quality", "improve risks prevention", "increase the resident's autonomy"and "ensure the individual and collective rights"*, or the structure performance like *"improve the opening of the structure to its environment"* and *"improve the structure's security"* or both.

- *Actions*

  The primary mission of a social and medico-social structure is, on the one hand, to take care of its residents and provide them with welfare and, on the other hand, to ensure well functioning of the structure, with respect to the laws and persons. Therefore, the actions of an action plan always aim to improve resident's life quality and the structure performance in general. Some examples of actions of the first aspect are: "*facilitate moving for persons with reduced mobility*", "*arrange trips*", "*organize shows*"or "*implement activities for the residents*". For the second aspect, typical actions are: "*train staff to use IT tools*", "*prepare a care protocol*"or "*install a lift*", for instances.

  The origins of the actions are either recommended by the structure assessment or decided by the managers for continuous improvements.

- *Constraints*

  There are two main hard constraints which must be always satisfied: the budget of the project and the minimum threshold that should be reached for each objective after completion of the action plan.

We note that the realized project is evaluated in the next assessment and each evaluation can lead to another project, and so on (see figure 1).

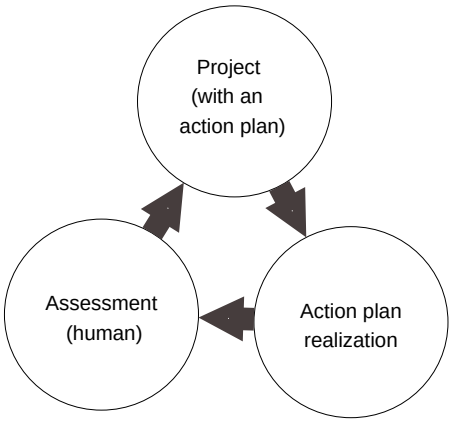

Figure 1: Assessment process in social and medico-social structures in France.

During the project elaboration, the decision maker is usually confronted with two main challenges: i) which actions should be implemented in order to maximize the overall quality of the structure while attaining the pre-defined objectives of the project; ii) which actions to choose when the project budget does not allow the implementation of all actions.

To assist the decision maker in elaborating the action plan, an action warehouse is incorporated to the "*MSQualité*"software to store all the feasible actions improving the structures quality, considering that the cost of each action as well as their impact on the objectives are known. The action warehouse is fed with actions already realized in different structures. When elaborating a project, the decision maker has just to define the objectives and possibly the corresponding budget. To help the decision maker in this task, "*MSQualité*"provides an action plan to achieve or improve the objectives which could be completed with new actions. The second challenge, which is the main issue of this work, is to determine an efficient action plan considering a limited budget: which actions to choose so that all the objectives are optimized without exceeding the allocated budget. The schema below (figure 2) illustrates the project elaborating process including the action plan optimization step.

Let us note that the realization step has the same duration as the project and could require scheduling tools to optimize the action plan realization according to the structure means. Nevertheless in this work we are not concerned by this aspect. Also, the assessment step allows analyzing the project results and draw conclusions and recommendations to take into account in the next project elaboration. On the other hand, according to the assessment results, some new actions (among those defined by the decision maker) could be added to the action warehouse which can be suggested to other similar structures and actions already stored in the action warehouse could be updated with the new results.

Thus, the problem consists in elaborating an efficient action plan to achieve a project's objectives while respecting the constraints. In other words, the aim is to choose a subset of actions among many possible actions while optimizing several often conflicting objectives under the given constraints. Each action has a realization cost and can influence (positively or negatively) a part or all of the objectives.

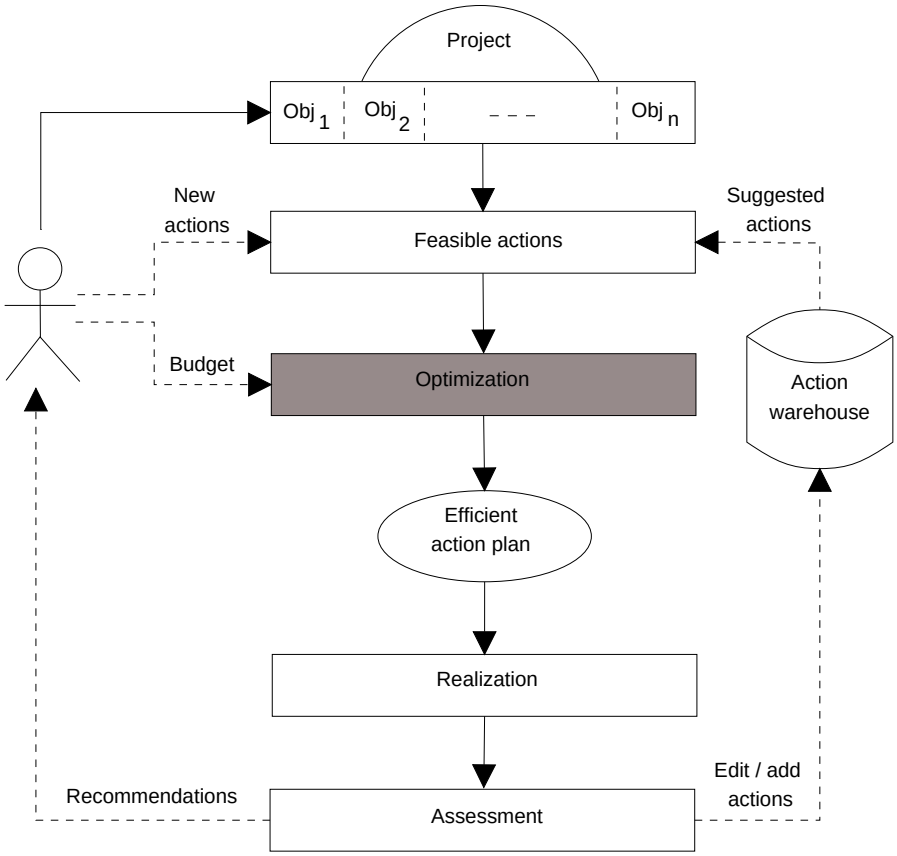

Figure 2: Project elaborating process.

2.2. Problem formulation

Before presenting the action plan problem formulation, let us recall the basic definition of the knapsack problem (KP) (Kellerer et al., 2004; Martello and Toth, 1990; Changdar et al., 2015) and its multiobjective version. Given n items each having a characteristic ω_i $(i = 1, ..., n)$ (weight, volume, cost, etc) and a profit p_i, some items should be selected to maximize the total profit without exceeding the capacity W of the knapsack. Formally:

$$(KP) \begin{cases} \max & \sum_{i=1}^{n} p_i x_i \\ \text{s.t.} & \sum_{i=1}^{n} \omega_i x_i \leq W \\ & x_i \in \{0,1\} \qquad i = 1, ..., n \end{cases} \tag{1}$$

where $x_i = 1$ means that the item i is selected to be in the knapsack, $x_i = 0$ otherwise.

From this basic formulation, several variants of the KP have been proposed and studied during the last decades: the multidimensional knapsack problem (MKP) (Kong et al., 2015; Puchinger et al., 2010; Wang et al., 2012), the multiple knapsack problem (Kellerer et al., 2004; Martello and Toth, 1990), the multiple-choice knapsack problem (MCKP) (Sinha and Zoltners, 1979) and the quadratic knapsack problem (QKP) (Gallo et al., 1980; Pisinger, 2007). Moreover, problems that combine two or more of these variants are also proposed, like the multiobjective multidimensional knapsack problem (Lust and Teghem, 2012; Vianna and Vianna, 2013) and the multidimensional multi-choice knapsack problem (Chen and Hao, 2014; Cherfi and Hifi, 2009; Shojaei et al., 2013). Other knapsack variants include for instance the constrained knapsack problem in fuzzy environment (Changdar et al., 2015) and dynamic multidimensional knapsack problems (Baykasoğlu and Ozsoydan, 2014).

The multiobjective knapsack problem is a difficult variant of the KP because the optimization of conflicting objectives simultaneously requires the use of specific techniques borrowed from the multiobjective research community. The MOKP may be used to formulate many practical problems like resource allocation, capital budgeting, project management and, in our case, action plan optimization.

Given a capacity constrained knapsack and n items (objects) having m profits f_{ij} $(j = 1, ..., m)$ and a characteristic ω_i. The MOKP consists of selecting a subset of items in order to maximize m profits (to maximize a vector of objective functions) while not exceeding the knapsack capacity W. Formally, the MOKP can be formulated as follows:

$$(MOKP) \begin{cases} \max & f_j(x) = \sum_{i=1}^{n} f_{ij} x_i & j = 1, ..., m \\ \text{s.t.} & \sum_{i=1}^{n} \omega_i x_i \leq W & \\ & x_i \in \{0, 1\} & i = 1, ..., n \end{cases} \tag{2}$$

where f_j represents the j^{th} objective function, m the number of the objectives, f_{ij} the value of item i on objective j, ω_i the weight of the item i. Most studies on the MOKP focus on instances of the bi-objective case ($m = 2$). Some studies are dedicated to the three-objective case ($m = 3$). Very few MOKP studies concern more objectives (Vianna et al., 2014; Vianna and Vianna, 2013).

Throughout this paper, the set of feasible actions is denoted by A, the action plan (which is a subset of actions selected in A) is denoted by a vector $x = (a_1, a_2, ..., a_n)$ where n refers to the size of A. Let $a_i = 1$ if the action a_i is selected and $a_i = 0$ otherwise. X denotes the search space (the set of all the possible action plans), F represents the set of the objectives $(f_1, f_2, ..., f_m)$ to maximize and m the number of objectives. Each objective is represented by a function f_j which associates to each action $a_i \in A$ its impact on the objective j. This impact is evaluated by the function $f_j(a_i) = v_{ij}$ which assigns to any action a_i an integer value v_{ij} reflecting the contribution of the action a_i to the achievement of the objective j (v_{ij}>0) or the degradation of the action a_i

for the objective j (v_{ij}<0). v_{ij}=0 indicates that action a_i has no effect on the objective j. Thus, we can associate to each action a_i an objective vector $V_i = (f_1(a_i), f_2(a_i), ..., f_m(a_i))$. The realization cost of an action a_i is denoted by ω_i which can take negative values, since there may be actions with negative costs when it is about selling objects or services, for instance. Considering an action plan $x = (a_1, a_2, ..., a_n) \in \{0,1\}^n$, the impact that x has on the objective j is given by:

$$f_j(x) = \sum_{i=1}^{n} a_i f_j(a_i) \tag{3}$$

and the global cost of x is given by:

$$C(x) = \sum_{i=1}^{n} a_i \omega_i \tag{4}$$

Also, we consider in this paper two constraints: (i) the global cost of the solution should not exceed a predefined budget β, i.e., $C(x) \leq \beta$. (ii) a constraint t_j is added for every objective j, determining the minimal threshold accepted for f_j, i.e., $f_j(x) \geq t_j$.

So, the optimization goal is to find $x^* \in \arg\max_{x \in X} F(x)$ verifying:

$$\begin{cases} x^* \in \{0,1\}^n \\ \forall j \in \{1,m\}, f_j(x^*) \geq t_j \\ C(x^*) \leq \beta \end{cases} \tag{5}$$

Since here F refers to a vector of evaluation functions, x^* is likely to be not unique. Instead, optimizing F leads to a set of mutually nondominated solutions called Pareto set (in Pareto optimality sense). We aim here to approximate the Pareto front effectively (the Pareto front refers to the image of the Pareto set in the objective space).

3. Binary quality indicators

This section is dedicated to binary quality indicators. We start with a general introduction followed by a presentation of the Epsilon and $R2$ indicators which are used in this work.

3.1. General presentation of binary quality indicators

In the following, X denotes the decision space of a general optimization problem, Z the corresponding objective space, and m the number of objective functions $f_1, f_2, ..., f_m$ that assign to each decision vector $x \in X$ a corresponding objective vector $z = \{f_1(x), f_2(x), ..., f_m(x)\} \in Z$. Given two solutions x_1 and x_2, the relation $x_1 \succ x_2$ means that solution x_1 is *preferable* to (or better than) solution x_2 according to the Pareto dominance relation (Zitzler et al., 2003). The set of all Pareto optimal solutions is called "Pareto optimal set ". The set

of all Pareto set approximations is represented by the symbol Ω. In this section, we assume that $Z \subseteq R^m$ and that m objectives are to be minimized.

The binary quality indicators (Zitzler et al., 2003) represent a natural extension of the Pareto dominance relation on sets of objective vectors, where a function $I : \Omega \times \Omega \longrightarrow \mathbb{R}$ quantifies the difference in quality between two approximation sets A and B. When R denotes a fixed reference set (e.g. the set of Pareto optimal solutions), the function I represents a unary quality indicator that assigns to each Pareto set approximation a real number which has to be minimized, to be as close as possible to R. With this principle, the optimization goal is transformed to the identification of a Pareto set approximation that minimizes I. Also, the quality indicator could be used to compare the quality of two single solutions or a single solution against a population and (with that comparison) it could be used in the selection process of evolutionary algorithms. Indeed, during the selection process, the solution to delete should be the one with the worst value of the indicator being used according to rest of the population. As indicated by the authors of IBMOLS (Basseur et al., 2012), "*During the selection process, the main objective is to remove the solutions which correspond to the smallest degradation of the overall quality of the population, in terms of the quality indicator being used*".

In the following, we define two indicators used in this study: the Epsilon additive indicator (I_ϵ) and the $R2$ indicator (I_{R2}). These definitions are taken partially from (Zitzler and Künzli, 2004; Zitzler et al., 2003) for Epsilon indicator and (Brockhoff et al., 2012; Hansen and Jaszkiewicz, 1998) for $R2$ indicator.

3.2. Epsilon indicator

$I_\epsilon(A, B)$ gives the minimum distance by which a Pareto set approximation A needs to be translated in each dimension of the objective space such that the approximation B is dominated by A. Formally, it can be defined as follows:

$$I_\epsilon(A, B) = min_\epsilon\{\forall x_2 \in B, \exists x_1 \in A : f_j(x_1) - \epsilon \leq f_j(x_2)\} \quad \forall j \in \{1, ..., m\} \tag{6}$$

I_ϵ can be used to compare the quality of two single solutions x_1 and x_2:

$$I_\epsilon(x_1, x_2) = max\{f_j(x_1) - f_j(x_2)\} \; \forall j \in \{1, ..., m\} \tag{7}$$

$I_\epsilon(x_1, x_2)$ represents the minimal translation to execute x_1 in the objective space so that it dominates x_2. This translation takes negative values when $x_2 \succ x_1$.

3.3. R2 indicator

The $R2$ indicator (Brockhoff et al., 2012; Hansen and Jaszkiewicz, 1998) and the Hypervolume (HV) indicator (Zitzler and Thiele, 1999) are two recommended approaches (Knowles et al., 2005) which simultaneously evaluate all desired aspects of a Pareto front approximations. However, the $R2$ indicator is considered as an alternative for the HV indicator for two reasons: (i) the run time of the HV is expected to increase exponentially with respect to the number

of objectives as its computation is known to be NP-hard. (ii) the distributions obtained using the HV are biased towards the knee regions of the Pareto front (Brockhoff et al., 2014).

As mentioned in (Hansen and Jaszkiewicz, 1998), the R indicator family is based on utility functions which map a vector $\vec{z} \in R^m$ to a scalar utility value $u \in \mathbb{R}$ for assessing the relative quality of two Pareto front approximation sets.

Definition 1. *For a discrete and finite set U of utility functions, a uniform distribution p over U, and a reference set R, the $R2$ indicator is defined by:*

$$R2(R, A, U) = \frac{1}{|U|} \sum_{u \in U} \left(\max_{r \in R}\{u(r)\} - \max_{a \in A}\{u(a)\} \right) \tag{8}$$

When R is constant, the $R2$ indicator can be defined as a unary indicator:

$$R2(A, U) = -\frac{1}{|U|} \sum_{u \in U} \max_{a \in A}\{u(a)\} \tag{9}$$

As suggested by Hansen and Jaszkiewicz (Hansen and Jaszkiewicz, 1998) and used by Brockhoff et al. in (Brockhoff et al., 2012) and (Brockhoff et al., 2014), we use, throughout this paper, the standard weighted Tchebycheff function $u(z) = u_\lambda(\vec{z}) = -\max_{j \in \{1,...,m\}} \lambda_j |z_j^* - z_j|$ within the $R2$ indicator as defined in equation 9 , where $\lambda = (\lambda_1, ..., \lambda_m) \in \Lambda$ is a given weight vector and z^* is an utopian point.

In (Brockhoff et al., 2012, 2014), Brockhoff et al. studied the influence of the number and the distribution of the weight vectors on the optimal distribution of the solutions. They concluded that: (i) For bi-objective problems, the optimal placement of a point according to the $R2$ indicator only depends on its neighbors and only on a subset of weight vectors. (ii) For a solution set of size μ, the optimal μ-distributions[2] for the $R2$ indicator turns out to contain the intersection points of the rays corresponding to the weight vectors with the Pareto front when $\mu > |\Lambda|$. There is one optimal μ-distribution when $\mu = |\Lambda|$ and the optimal μ-distributions is not always unique in the case $\mu < |\Lambda|$. (iii) Even in the scenarios in which the optimum distribution changes with increasing the number of the weight vectors $|\Lambda|$, this distribution stabilizes when a specific number $|\Lambda|_{threeshold}$ is exceeded. For the bi-objective case, $|\Lambda|_{threeshold} = 10\mu$.

The same authors show in (Wagner et al., 2013) that the optimal μ-distributions can be affected by moving the reference point, by restricting the weight space and by skewing the weight vectors distribution (Figure 3).

[2]Optimum sets of size μ.

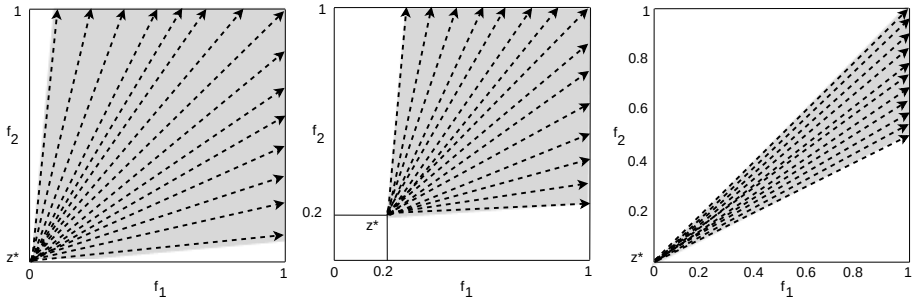

Figure 3: Moving the target cone (gray area) by changing the position of the reference point from $z^* = (0, 0)$ (left) to $z^* = (0.2, 0.2)$ (center). Narrowing the target cone (gray area) by restricting the first component of the normalized weight vectors from [0.1,0.9] (left) to [0.5,0.7] (right). The dashed arrows correspond to the target directions defined by the weight vectors.

4. Indicator-based multiobjective local search (IBMOLS)

In the field of evolutionary multiobjective optimization, the use of indicator-based algorithms is continuously increasing, especially since 2004. In this section, we review some representative studies on indicator-based multiobjective optimization, then we present the indicator-based multiobjective local search approach applied to the action plan optimization problem.

4.1. Indicator-based multiobjective optimization

Many multiobjective optimization algorithms from the literature are based on the use of evolving a population of solutions to find a good set of compromise solutions. The binary quality indicator principle was first proposed within an evolutionary multiobjective (EMO) algorithm in (Zitzler and Künzli, 2004). Since then, several other methods and studies using quality indicators have emerged such as: an EMO algorithm using the hypervolume measure as selection criterion (Emmerich et al., 2005), handling uncertainty in indicator-based multiobjective optimization (Basseur and Zitzler, 2006), improving hypervolume-based EMO algorithms by using objective reduction methods (Brockhoff and Zitzler, 2007), $R2$-IBEA (Phan and Suzuki, 2013), $R2$-EMOA (Trautmann et al., 2013) and $R2$ indicator-based multiobjective search (Brockhoff et al., 2014). Additionally, Bader and Zitzler (2011) proposed the HypE algorithm, a hypervolume estimation algorithm for multiobjective optimization that uses Monte Carlo simulation to approximate the exact hypervolume values. Their experimental results indicate that HypE is highly effective for many-objective problems. More recently, Jiang et al. (2015) introduced a simple and fast hypervolume indicator-based multiobjective evolutionary algorithm (FV-MOEA) to quickly update the exact HV contributions of solutions. They tested their algorithm on 44 benchmarks with 2-5 objectives and demonstrated that FV-MOEA reports high hypervolume values and obtains significant speedups compared to other HV indicator-based MOEAs. Ben Mansour and Alaya (2015) proposed an

indicator-based ant colony optimization algorithm for multiobjective knapsack problem. Li et al. (2015) developed a general multiobjective particle swarm optimizer based on the $R2$ indicator (called $R2$-MOPSO). $R2$-MOPSO uses $R2$ contribution of the archived solutions to select global best leaders to update the swarm. Falcón-Cardona and Coello (2016) presented a new indicator-based multiobjective ant colony optimization algorithm for continuous search spaces (iMOACO$_{\mathbb{R}}$), which is based on the ACO$_{\mathbb{R}}$ and $R2$ indicators. Díaz-Manríquez et al. (2016) combined the $R2$ indicator and particle swarm optimization to solve multi/many-objective problems.

On the other hand, local search algorithms are also known to be efficient for many real-world applications, and especially on large-scale problems. IBMOLS (Basseur and Burke, 2007) is a multiobjective algorithm combining a quality indicators and a local search mechanism. Contrary to most multiobjective local search algorithms of the literature which are based either on the Pareto dominance relation between solutions or on aggregation methods, the fitness assignment in IBMOLS is based on the quality indicator principle. It does not require any specific diversity preservation mechanism, since this aspect should be considered in the indicator itself. Moreover, the decision maker can include preferences in the indicator definition. Furthermore, the local search deals with a fixed population size, which enables the algorithm to find multiple nondominated solutions in a single run, without any specific mechanism dedicated to control the number of nondominated solutions during the local search process (problem encountered with the classical multiobjective local search "Pareto Local Search"(Paquete et al., 2004)). Finally, IBMOLS requires only a small number of parameters.

Several studies of the literature have shown the efficiency of the IBMOLS method on different multiobjective problems. For instance, Basseur et al. (2012) applied IBMOLS to three different problems (Flow-Shop, Ring Star and Nurse Rostering). Then, in (Basseur et al., 2012), the authors proposed HBMOLS that uses the hypervolume contribution unary indicator I_{HC} in the selection process. Tangpattanakul et al. (2015) applied IBMOLS with the hypervolume indicator to solve a multiobjective optimization problem associated with selecting and scheduling observations of an agile Earth observing satellite. They compared IBMOLS with the BRKGA algorithm (Gonçalves and Resende, 2011) and showed the advantage of IBMOLS over BRKGA. Finally, Tangpattanakul (2015) also used IBMOLS with the hypervolume indicator to solve a bi-objective p-Median problem and showed its effectiveness.

4.2. Indicator-based multiobjective local search for action plan optimization

Let P denote the current population of solutions of the IBMOLS algorithm. Then, a local search procedure is applied on P as described below. A local search step of the IBMOLS algorithm corresponds to a local search step applied on each solution in P. A neighbor is accepted if its indicator value is better than the worst solution in P. The neighborhood generation stops when the entire neighborhood of solution in P is explored or once an improving solution is found (first neighboring solution that improves the quality of P with respect

to I). As such, the neighborhood is not explored entirely for seeking the best neighbor. Two main reasons guided this choice: (i) it often enables to speed up the convergence of the population, since most of the time only a small part of the neighborhood is generated. (ii) The selection of the best neighbor leads to deterministic local search steps (one possible way to go from an initial solution to a set of local optima). The selection of a random improving move allows us to reach different local optima (in the sense of multiobjective optimization) from a single initial solution. The entire local search is terminated when the archive E of nondominated solutions has not received any new solution during a complete local search step.

Moreover, it is assumed that objective values of all solutions are normalized; to achieve this, the minimum m_j and maximum M_j value of each objective function f_j in the population P are computed first:

$$\begin{cases} m_j = min_{x \in P}(f_j(x)) \\ M_j = max_{x \in P}(f_j(x)) \end{cases} \tag{10}$$

Then each objective function j of every individual x is normalized as follows:

$$NF_j(x) = \frac{f_j(x) - m_j}{M_j - m_j} \tag{11}$$

where $NF_j(x)$ is the normalized j^{th} objective function of the individual x. The extreme values of the population are computed after the initialization process and after each local search step (see algorithm 1). To compute an indicator value $I(x_1, x_2)$, the normalized values of objective functions $NF_j(x)$ are employed. A detailed description of IBMOLS is outlined in algorithm 1.

In the iterated version of IBMOLS (algorithm 2), a Pareto set approximation PO is maintained and updated. After each local search, a new initial population is created for the next IBMOLS execution, using the *initWalk* function.

Note that IBMOLS is defined only by two main parameters (the population size N, the binary indicator I) and the function *initWalk* which initializes the population. These parameters can be tuned dynamically during the search or fixed according to the problem instance under consideration.

The population size of IBMOLS is fixed and does not depend upon the problem, objective functions or space dimension being considered. Basseur et al. (2012) have shown on three different problems (a Flow-Shop problem, a Ring Star problem and a Nurse Scheduling problem) that IBMOLS performs well especially when using a small population (not more than 15 individuals). Based on this conclusion, we have used populations of 10 solutions in our experiments.

The choice of an appropriate quality indicator, according to the considered problem, is primordial to obtain a good approximation of the Pareto front. In (Basseur et al., 2012), the efficiency of two binary indicators (I_ϵ and I_{HV}) was compared against I_{Ben}, I_{Fon} and I_{Sri}, which are derived from classical ranking methods based on the Pareto dominance relation. It is concluded that, for the bi-objective case, the I_ϵ and I_{HV} indicators tend to outperform the Pareto dominance based indicators, especially when the problem size increases, I_ϵ being

Algorithm 1 The IBMOLS algorithm

Input: P (initial population of size N); I (binary indicator)
Output: E (Pareto approximation set)
Step 1: $E \longleftarrow nondominatedSolutions(P)$
Step 2 - fitness assignment: Calculate fitness values of individual x in P, i.e., $fit(x) = I(P \backslash \{x\}, x)$
Step 3 - local search step: For all $x \in P$ do:
update, for each objective function f_j, the minimal m_j and maximal M_j values in P (for objective functions normalisation)
repeat
1) $x^* \longleftarrow$ one unexplored neighbor of x
2) $P \longleftarrow P \cup x^*$
3) compute x^* fitness: $I(P \backslash \{x^*\}, x^*)$
4) update all $z \in P$ fitness values: $fit(z) += I(x^*, z)$
5) $w \longleftarrow$ the worst individual in P
6) remove w from P
7) update all $z \in P$ fitness values: $fit(z) -= I(w, z)$
until all neighbors are explored or $w \neq x^*$
Step 4 - termination: $E \longleftarrow nondominatedSolutions(E \cup P)$. If E does not change, then return E; else perform another local search step.

Algorithm 2 Iterated IBMOLS algorithm

Input: N (population size) ; I (binary indicator)
Output: PO (Pareto approximation set)
Step 1: $PO \longleftarrow \varnothing$
Step 2: while stopping condition not achieved do
1) $P \longleftarrow initWalk(PO, N)$
2) $E \longleftarrow$ IBMOLS(P, I)
3) $PO \longleftarrow nondominatedSolutions(PO \cup E)$
Step 3: Return PO

globally more efficient. In (Basseur et al., 2012), HBMOLS has been proposed as a modified version of IBMOLS. HBMOLS uses the hypervolume contribution unary indicator I_{HC} to define the selection process of a HBMOLS-based iterated local search. Experiments showed the superiority of HBMOLS over different IBMOLS versions. Nevertheless, it is known that I_{HC} is very expensive to compute when the number of objectives is more than 2. On the other hand, we have noted in (Chabane et al., 2015) that with I_ϵ the diversity is reduced. Moreover, it is reported in (Wessing and Naujoks, 2010) that the hypervolume and $R2$ indicators show a correlated behavior, in practice. For these reasons, in the experiments presented is this paper, we have used, in addition to I_ϵ, the $R2$ indicator.

Even if the initial population is entirely created randomly for the first iteration, when we iterate the local search process, the $initWalk$ function generates a new population P for the next iteration using information about good solutions obtained during the previous iterations. Indeed, the $initWalk$ function applies random mutations on N randomly selected solutions of PO (each solution of PO can only be selected at most once). To each selected solution, the mutation is applied with a probability of $1/n$ (where n is the number of actions) and the mutated solution is added to the population if it is not present in the population and if it additionally verifies the budget constraint β and the objective thresholds. When $|PO| < N$, all solutions of PO are selected and the missing individuals of P are filled with new random solutions.

5. Experimental setup

In this and next sections, we present a set of experiments aiming to allow a comparison of the results obtained with $R2$-IBMOLS, ϵ-IBMOLS and NSGA-II (Deb et al., 2002) applied to the action plan optimization problem. Before discussing the experiment protocol and providing the results (Section 6), we first explain how the test instances were generated.

5.1. Instances generation

Based on the action plan optimization model given in section 2.2 and a study of ten real action plans, we have generated several partially structured instances with different actions, $n \in \{50, 100, 150, 250, 500\}$ and different number of objectives, $m = 2, 3, 4, 5, 6, 8$. To be as close as possible with respect to the real action plans, for each objective function, an action has a chance of 50% to be neutral, 40% to have a positive impact and 10% to have a negative impact. Moreover, the cost of 40% of the actions is set to zero. The non-null action values are uniformly taken from the interval [0,100] (positively or negatively). The non-null action costs are uniformly taken in the interval $[-10^4, 10^4]$. The instances used in our experiments are available at: `http://www.info.univ-angers.fr/pub/hao/gepiplanning/R2-IBMOLS.zip`.

5.2. Experimental protocol

Using the instances above, we have tested $R2$-IBMOLS, ϵ-IBMOLS and NSGA-II with the commonly used parameters in the literature. For NSGA-II, we have used a population of size 100, a mutation probability of $1/n$ (where n is the number of the actions). For ϵ-IBMOLS and $R2$-IBMOLS we have used the iterative version with a population of size 10. For a current population P, a reference point z^* and a set of weight vectors Λ, the fitness of each solution x in P is evaluated, with respect to the rest of the population, using the formula 12 for ϵ-IBMOLS and the formula 13 for $R2$-IBMOLS.

$$I_\epsilon(P\backslash\{x\}, x) = min_{z \in P\backslash\{x\}}(I_\epsilon(z, x)) \tag{12}$$

$$I_{R2}(x, \Lambda, z^*) = I_{R2}(P, \Lambda, z^*) - I_{R2}(P\backslash\{x\}, \Lambda, z^*) \tag{13}$$

$R2$-IBMOLS is used with the reference point $z^* = (2, 2, ..., 2)$ and 100 weight vectors ($|\Lambda| = 100$), uniformly distributed in the objective space. To generate these vectors, we have used algorithm (3), initially proposed by Suzuki and Phan (Phan and Suzuki, 2013).

This algorithm uses the hypervolume indicator to produce weight vectors so that they uniformly disperse and maximize their hypervolume in the objective space. This method is interesting because it does not depend on the dimension of the objective space and works in the same way for low-dimensional and high-dimensional spaces. This method can, however, be time consuming if the weight vectors are generated at each iteration. In our experiments, the weight vectors are generated once and remain the same throughout each experiment.

For the three algorithms ($R2$-IBMOLS, ϵ-IBMOLS and NSGA-II), the initial population is generated randomly while satisfying the following constraints: i) the costs of the individuals do not exceed the budget β; ii) each individual x should improve all the objectives ($f_j(x) > 0 \; \forall j \in \{1, ..., m\}$). Algorithm 4 shows the initial population generation procedure.

Also, a bounded population size is used and the following selection strategy is adopted: one random neighbor of each individual of the current population is selected to be a member of the child population in NSGA-II or to integrate the current population of IBMOLS. The neighborhood generation remains unchanged for the three methods: the i^{th} neighbor of the solution $x = (a_1, a_2, ..., a_n)$ is obtained by flipping the value of a_i and only the neighbors verifying the constraint β and the objective thresholds are considered as candidate (when the cost of the neighbor is greater than β, another neighbor is generated). For all instances, the budget constraint β is fixed to one million and the thresholds are fixed to 1 ($t_j \geq 1 \; \forall j \in \{1, ..., m\}$).

For the quality assessment, we have performed 30 runs of each method on each instance. The stopping condition for each run corresponds to $200 * n * m$ evaluated solutions (where n is the number of actions and m is the number of objectives). The experiments are realized on an Intel core i5-2400 CPU machine with 2 x 3.10Ghz and 16Gb of RAM.

Algorithm 3 Generation of weight vectors

Input: t_{max}, the maximum number of iterations
Input: l, the number of weight vectors to be generated
Input: m, the number of objectives
Output: Λ, the set of weight vectors generated

```
t = 0
Λ = ∅
while t < t_max do
    Randomly choose a vector σ in [0,1]^{m-1} : σ = (σ_1, σ_2, ..., σ_{m-1})
    Sort σ_j in σ ascending order, such that σ_1 ≤ σ_2 ≤ ... ≤ σ_{m-1}
    Create a vector λ = (λ_1, λ_2, ..., λ_{m-1}, λ_m) = (σ_1, σ_2 − σ_1, ..., σ_{m-1} −
    σ_{m-2}, 1 − σ_{m-1})
    Λ = Λ ∪ {λ}
    if |Λ| > l then
        Calculate hypervolume contribution of each λ in Λ:
        I_HC(λ) = HV(Λ) − HV(Λ \ {λ})
        λ* = argmin_{λ∈Λ} I_HC(λ)
        Λ = Λ \ {λ*}
    end if
    t = t + 1
end while
```

Algorithm 4 Initial population generation

Input: N (population size) ; β (budget)
Output: P (initial population)

```
P = ∅
while |P| < N do
    x = randomSolution()
    while cost(x) > β do
        x = x less one random action with positive cost
    end while
    for each objective function f_j do
        while f_j(x) ≤ 0 do
            x = x less one random action with negative value for f_j
        end while
    end for
    P = P ∪ {x}
end while
Return P
```

For the statistical analysis, we have used the performance assessment package PISA, described in (Knowles et al., 2005). We have performed a pair-wise comparison of the algorithms for each instance using the Mann-Whitney test and applied the Bonferroni correction to adjust the individual significance levels. Under the Bonferroni correction, in order to achieve a significant level of l, the pair-wise tests should be performed with a significant level of $\frac{l}{h}$, where h is the number of the pair-wise comparison, which is three in this study. Using the R and the *Epsilon* indicators, we obtain the p–*value* corresponding to the hypothesis "*the first algorithm performs better than the second one*". This corresponds to the lowest significance level for which the null-hypothesis is rejected. In our experiments, we say that algorithm A_1 outperforms algorithm A_2 if the Mann-Whitney test provides the conclusion "A_1 *performs better than* A_2" with a confidence level greater than 95% (p–$value \leq 0.05$), resulting in an individual significance level of 0.0167, after applying the Bonferroni correction. The test procedure and the performance assessment package PISA are available at the following URL: `http://www.tik.ee.ethz.ch/sop/pisa/?page=assessment.php`.

6. Computational results

In this section, we present the experimental results obtained on the simulated data with the experimental protocol described previously.

In the practical case, it is recommended to present a reduced number of high-quality solutions to the decision maker. Indeed, one solution could have several tens of actions and the decision maker should be able to choose the most adequate action plan easily. Table 2 reports the average number of nondominated solutions obtained with each algorithm over 30 runs. We note that the number of obtained solutions is highly variable for the three algorithms, ranging from several tens of solutions for small size instances to several thousands of solutions for large size instances. Even for the smallest instance ("2_50"), the number of solutions may be high so that the decision maker cannot make the choice easily (32 solutions for R2-IBMOLS, 47 solutions for ϵ-IBMOLS and 2,150 solutions for NSGA-II). Hence it would be interesting to integrate decision maker preferences in the optimization process to direct the research process towards a specified region of the objective space to obtain just the most interesting solutions for the decision maker (Ruiz et al., 2015). Preference elicitation methods (Chen and Pu, 2004), especially those based on weight elicitation strategies (Benabbou and Perny, 2015), could be a good mean to this end.

Table 1 reports the comparison between NSGA-II, ϵ-IBMOLS and R2-IBMOLS in terms of mean values obtained for the R and ϵ indicators over 30 runs, using the set of 30 instances of different sizes. The first column presents the instance name, indicating its main characteristics: m and n respectively correspond to the number of objectives and the number of actions considered. The values in bold mean that the corresponding algorithm is at least 95% statistically better than the two others for the considered instance and indicator. The underlined values mean that the correspond algorithm is at least 95% statistically better than the algorithm corresponding to the values in normal style, but

Table 1: Comparison of mean values of the ϵ and R indicators for NSGA-II, ϵ-IBMOLS and $R2$-IBMOLS.

Instance	**Assessment with ϵ indicator**			**Assessment with R indicator**		
	NSGA-II	**ϵ-IBMOLS**	**R2-IBMOLS**	**NSGA-II**	**ϵ-IBMOLS**	**R2-IBMOLS**
2_50	0.323	0.567	**0.144**	0.118	0.310	**0.027**
2_100	0.314	0.549	**0.105**	0.132	0.303	**0.018**
2_150	0.346	0.577	**0.124**	0.144	0.279	**0.023**
2_250	0.335	0.524	**0.126**	0.144	0.279	**0.028**
2_500	0.325	0.619	**0.116**	0.149	0.334	**0.030**
3_50	0.296	0.425	**0.171**	0.068	0.080	**0.022**
3_100	0.315	0.433	**0.175**	0.085	0.117	**0.026**
3_150	0.365	0.413	**0.181**	0.090	0.139	**0.030**
3_250	0.432	0.376	**0.118**	0.145	0.074	**0.021**
3_500	0.532	0.261	**0.155**	0.148	0.075	**0.032**
4_50	0.261	0.350	**0.174**	0.044	0.059	**0.016**
4_100	0.466	0.192	**0.086**	0.083	0.034	**0.009**
4_150	0.497	0.248	**0.191**	0.080	0.050	**0.026**
4_250	0.408	0.118	**0.086**	0.035	0.021	**0.009**
4_500	0.571	0.122	**0.111**	0.108	0.021	**0.014**
5_50	0.227	0.366	**0.135**	0.032	0.042	**0.011**
5_100	0.399	0.208	**0.169**	0.052	0.030	**0.014**
5_150	0.446	0.191	**0.129**	0.068	0.028	**0.012**
5_250	0.505	0.089	**0.039**	0.096	0.011	**0.003**
5_500	0.381	0.068	**0.032**	0.074	0.009	**0.003**
6_50	0.232	0.348	**0.167**	0.031	0.052	**0.014**
6_100	0.486	0.144	**0.092**	0.057	0.012	**0.004**
6_150	0.510	0.127	**0.079**	0.084	0.013	**0.004**
6_250	0.514	0.091	**0.054**	0.057	0.009	**0.003**
6_500	0.556	0.153	**0.065**	0.089	0.015	**0.006**
8_50	0.158	0.286	**0.122**	0.013	0.015	**0.005**
8_100	0.227	0.105	**0.084**	0.008	0.006	**0.002**
8_150	0.246	0.077	**0.062**	0.005	0.004	**0.001**
8_250	0.271	0.05	**0.041**	0.004	0.003	**0.001**
8_500	0.436	0.225	**0.076**	0.073	0.021	**0.009**

it is worst than the algorithm corresponding to the values in bold style, for the considered instance and indicator.

From Table 1, we can conclude that $R2$-IBMOLS is quite efficient on all the instances. NSGA-II is better than ϵ-IBMOLS on the small instances (all the instances with 2 objectives, all the instances with 50 actions and the instances 3_100 and 3_150). Meanwhile, ϵ-IBMOLS performs better than NSGA-II as soon as the number of objectives exceeds 4.

Table 2: Average number of nondominated solutions generated by NSGA-II, ϵ-IBMOLS and $R2$-IBMOLS over 30 runs.

Instance	**NSGA-II**	**ϵ-IBMOLS**	**R2-IBMOLS**
2_50	2,150	47	32
2_100	1,529	66	48
2_150	1,450	66	67
2_250	1,300	86	89
2_500	1,279	119	135
3_50	2,528	210	333
3_100	2,564	270	629
3_150	3,061	335	845
3_250	3,947	495	1,696
3_500	5,165	703	2,856
4_50	585	87	358
4_100	1,009	164	671
4_150	1,131	235	905
4_250	1,143	305	2,348
4_500	1,769	663	3,558
5_50	1,148	215	522
5_100	1,979	426	1,447
5_150	2,114	502	1,949
5_250	3,046	598	3,266
5_500	4,243	1,058	9,501
6_50	1,542	341	651
6_100	2,828	697	2,097
6_150	3,952	908	3,725
6_250	5,075	1,210	7,727
6_500	7,687	2,492	18,900
8_50	3,991	965	3,989
8_100	7,988	2,151	4,918
8_150	11,989	3,080	5,317
8_250	19,991	4,919	19,176
8_500	39,996	9,325	45,419

For the action plan optimization problem studied in this work, the algorithm run time is not necessarily a main aspect for evaluating its effectiveness, since the project is done once every 5 years. Nevertheless, we note, from our experiments, that even if $R2$-IBMOLS is more efficient than NSGA-II and ϵ-IBMOLS (see Table 1), its run time should be reduced, especially for the large size instances where the run time of $R2$-IBMOLS is high. The run times obtained for some large size instances are (in seconds): 12,193.796 for the instance 3_500, 20,392.545 for the instance 4_500, 152,636.384 for the instance 5_500, 29,026.848

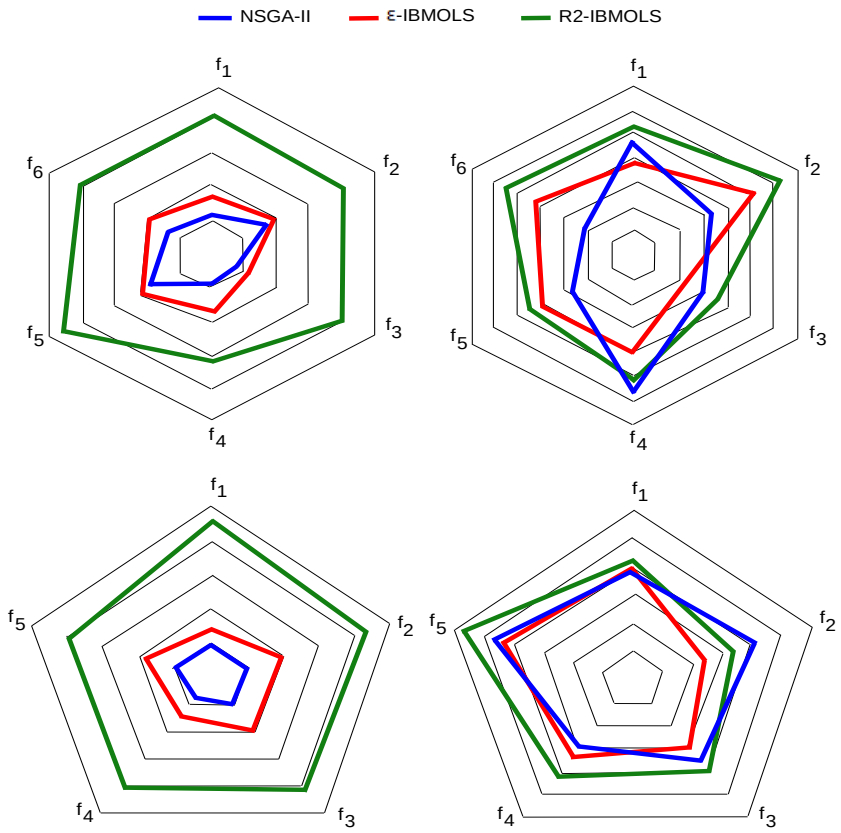

Figure 4: Maximum values obtained with NSGA-II, ϵ-IBMOLS and $R2$-IBMOLS over 30 runs, for objectives of four instances with 50 or 500 actions and 5 or 6 objectives: "6_500"(top left), "6_50"(top right), "5_500"(bottom left) and "5_50"(bottom right.)

for the instance 6_250 and 690,734.872 for the instance 6_500. Indeed, for all the instances with more than 2 objectives, the average run time of $R2$-IBMOLS is significantly higher than the one of ϵ-IBMOLS and NSGA-II. The run time of $R2$-IBMOLS essentially depends on the number of weight vectors being considered. It could be reduced by exploiting the following property of the $R2$ indicator: the optimal placement of solutions in the objective space only depends on a subset of the weight vectors (vectors around each solution).

Figure 4 reports the maximum values obtained on each objective function, with NSGA-II, ϵ-IBMOLS and $R2$-IBMOLS over 30 runs, on four representative instances of the action problem considered here (instances with 5 and 6 objectives). Indeed, in real-world instances the number of actions can vary from 50 actions to more than 500 actions and the number of objectives in our practical case can be greater than four. It is clear, from figure 4, that $R2$-IBMOLS attains more likely extreme solutions for each objective function than NSGA-II and ϵ-IBMOLS: on the four considered instances, the maximum values obtained for all

the objectives with $R2$-IBMOLS are higher than those obtained with NSGA-II and ϵ-IBMOLS except the objective f_2 of the instance "5_50"(bottom right) and the objective f_4 of the instance "6_50"(top right).

7. Practical impact on action plan elaboration

As indicated in Section 2.1, this work is part of the "*MSQualité*"software developed by the Company *GePI* for social and medico-social sector. As one key component of "*MSQualité*", the solution presented in this work provides valuable assistance to managers on the following aspects.

- Each structure manager can be aware of the actions already implemented in other structures and especially those actions leading to quality improvement of the structures. This information helps the manager to make better decisions concerning the actions to be taken in his own structure.
- To evaluate the real impact of the performed actions on the objectives, managers often use indicators that could be numerous and varied, depending on the nature of the actions. However, having a common platform in "*MSQualité*"allows each manager to exploit and benefit from indicator grids built by other structures to assess his own action plan.
- As mentioned in Section 2.1, one difficult task faced by managers is the choice of the most appropriate actions to implement. With the method proposed in this work, this task becomes easier since the manager has just to define the objectives of his project, and "*MSQualité*"will suggest actions whose positive impacts have already been proven in the past in at least one structure. In real action plans, about 60% to 80% of actions could be proposed by "*MSQualité*".
- With the proposed method, we estimate that, for an equal budget, managers could elaborate action plans which are 40% to 50% more efficient than those elaborated manually. Indeed, 40% to 50% of actions of real action plans are usually implemented without knowing beforehand whether the actions will be efficient or not, or to which extend they will be efficient. With the proposed method, those actions could be replaced by actions whose positive impacts are already known, since when a given action has already been implemented and evaluated in any structure, the results of that evaluation are accessible to all the managers. Hence, managers will be able to elaborate more efficient action plans.

8. Conclusion and perspectives

In this paper, we have studied an action plan optimization problem encountered in the fast evolving social and medico-social sector. The targeted application requires identification of a set of efficient actions among many candidate actions within a given social and medico-social structure while satisfying

some imperative constraints and optimizing a number of objectives. To tackle this problem, we have proposed a mathematical formulation of this application based on the multiobjective knapsack problem. As our solution method, we have adapted the IBMOLS approach with the ϵ and $R2$ indicators. We have assessed the proposed approach on simulated data with 50-500 actions and 2-8 objectives and performed comparative studies between IBMOLS (ϵ-IBMOLS and $R2$-IBMOLS) and the popular NSGA-II algorithm.

Based on this work, we can make two main conclusions. First, the adopted IBMOLS approach with the ϵ and $R2$ indicators proves to be a viable method to find high-quality compromise solutions for the action plan optimization problem in social and medico-social structures. As such, its integration within the decision support system "*MSQualité*" constitutes a valuable means which helps managers to elaborate more effectively efficient action plans to continually improve the quality of their structures. Second, thanks to the shared platform offered by "*MSQualité*", managers of different structures can collaborate with each other and benefit mutually from the experience of all structures.

This work also opens the way for future research. First, in this work, optimization is performed independently of decision maker preferences. However, the $R2$ indicator offers some properties to do that. Indeed, we can use the position of the reference point and the weight vector direction of the $R2$ indicator to shrink the search space and orientate the search process towards regions of interest of the decision maker in the objective space. This would also help to reduce the number of generated solutions and thus ease manager's decision making. Second, in our experiments, we have noticed that $R2$-IBMOLS whose run time depends on the number of the used weight vectors is more time consuming than ϵ-IBMOLS and NSGA-II on the large instances. This problem can be alleviated by considering the fact that the optimal placement of solutions according to the $R2$ indicator depends only on a subset of the weight vectors. Finally, this work has shown that the proposed approach works well for our studied problem whose objectives are limited to 8. An interesting work would be to investigate IBMOLS on problems with more objectives and higher complexity and compare its performance with other competitive multiobjective approaches such as HypE (Bader and Zitzler, 2011) and MOEA/D (Zhang and Li, 2007). Meanwhile, notice also that in the proposed approach, the adopted optimization algorithm (IBMOLS) can be substituted by other algorithms.

To conclude, this study shows that computerized decision support systems have much to offer in the fast evolving social and medico-social sector and can effectively help decision makers to identify suitable choices to continually improve the overall quality of their structures.

Acknowledgments

We are grateful to our reviewers for their insightful comments which helped us to improve the paper. This work was partially supported by the French Ministry for Research and Education through a CIFRE grant (number 0450/2013). We thank Rachid Naitali, the General Director of *GePI*, for his support.

References

Bader, J., and Zitzler, E. (2011). HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1), 45–76.

Barichard, V. and Hao, J.K. (2003). Genetic tabu search for the multi-objective knapsack problem. *Journal of Tsinghua Science and Technology*, 8(1), 8–13.

Basseur, M., and Burke, E. (2007). Indicator-based multi-objective local search. In Proceedings of *IEEE Congress on Evolutionary Computation (CEC 2007)*, pp. 3100–3107.

Basseur, M., Liefooghe, A., Burke, E., and Lee, K. (2012). The efficiency of indicator-based local search for multi-objective combinatorial optimisation problems. *Journal of Heuristics*, 18(2), 263–296.

Basseur, M., Zeng, R.Q., and Hao, J.K. (2012). Hypervolume-based multi-objective local search. *Neural Computing and Applications*, 21(8), 1917–1929.

Basseur, M., and Zitzler, E. (2006). Handling uncertainty in indicator based multiobjective optimization. *International Journal of Computational Intelligence Research*, 2(3), 255–272.

Baykasoğlu, A., and Ozsoydan, F.B (2014). An improved firefly algorithm for solving dynamic multidimensional knapsack problems. *Expert Systems with Applications*, 41(8), 3712–3725.

Benabbou, N., and Perny, P. (2015). Incremental weight elicitation for multiobjective state space search. In Proceedings of *Association for the Advancement of Artificial Intelligence Conference (AAAI'15)*, Austin USA.

Ben Mansour, I., and Alaya, I. (2015). Indicator Based Ant Colony Optimization for Multi-objective Knapsack Problem. *Procedia Computer Science*, 60, 448–457.

Brockhoff, D., Wagner, T., and Trautmann, H. (2012). On the Properties of the R2 Indicator. In *Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation*, GECCO '12, pp. 465–472.

Brockhoff, D., Wagner, T., and Trautmann, H. (2014). R2 indicator-based multiobjective search. *Evolutionary Computation*, 23(3), 369–395.

Brockhoff, D., and Zitzler, E. (2007). Improving hypervolume-based multiobjective evolutionary algorithms by using objective reduction methods. In *Congress on Evolutionary Computation*, pp. 2086–2093.

Chabane, B., Basseur, B., and Hao, J.K. (2015). A practical case of the multiobjective knapsack problem: Design, modelling, tests and analysis. In Clarisse Dhaenens, Laetitia Jourdan, and Marie-Eléonore Marmion, editors, *Learning and Intelligent Optimization*, number 8994 in Lecture Notes in Computer Science, pp. 249–255. Springer.

Changdar, C., Mahapatra, G.S., and Pal, R.K. (2015) An improved genetic algorithm based approach to solve constrained knapsack problem in fuzzy environment. *Expert Systems with Applications*, 42(4), 2276–2286.

Chen, Y., and Hao, J.K. (2014). A reduce and solve approach for the multiple-choice multidimensional knapsack problem. *European Journal of Operational Research*, 239(2), 313–322.

Chen, L., and Pu, P. (2004). Survey of preference elicitation methods. Technical Report IC/2004/67, Ecole Politechnique Federale de Lausanne, Switzerland.

Cherfi, N., and Hifi, M. (2009). Hybrid algorithms for the multiple-choice multi-dimensional knapsack problem. *International Journal of Operational Research*, 5(1), 89–109.

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

Díaz-Manríquez, A., Toscano, G., Barron-Zambrano, J.H., and Tello-Leal, E. (2016). R2-Based Multi/Many-Objective Particle Swarm Optimization. *Computational Intelligence and Neuroscience*, Article ID 1898527, 10 pages.

Emmerich, M., Beume, N., and Naujoks, B. (2005). An EMO algorithm using the hypervolume measure as selection criterion. Carlos A. Coello Coello, Arturo Hernndez Aguirre and Eckart Zitzler, editors, *Evolutionary Multi-Criterion Optimization*, volume 3410 of Lecture Notes in Computer Science, pp. 67–76. Springer.

Falcón-Cardona, J.G., and Coello, C.A.C. (2016). iMOACO$_{\mathbb{R}}$: A New Indicator-Based Multi-objective Ant Colony Optimization Algorithm for Continuous Search Spaces. In Handl et al. editors, *Parallel Problem Solving from Nature PPSN XIV*, volume 9921 of Lecture Notes in Computer Science, pp. 289–398. Springer.

Gallo, G., Hammer, P.L., and Simeone, B. (1980). Quadratic knapsack problems. In M. W. Padberg, editor, *Combinatorial Optimization*, number 12 in Mathematical Programming Studies, pp. 132–149. Springer Berlin Heidelberg.

Gonçalves, J.F, and Resende, M.G.C. (2011). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5), 487–525.

Hansen, M.P., and Jaszkiewicz, A. (1998). Evaluating the quality of approximations to the non-dominated set. Technical Report IMM-REP-1998-7, Institute of Mathematical Modelling Technical, University of Denmark.

Jiang, S., Zhang, J., Ong, Y.S,. Zhang, A.N., and Tan, P.S. (2015). A simple and fast hypervolume indicator-based multiobjective evolutionary algorithm. *IEEE Transactions on Cybernetics*, 45(10), 2202–2213.

Kellerer, H., Pferschy, U., and Pisinger, D. (2004). *Knapsack Problems*. Springer Science & Business Media.

Kong, X., Gao, L., Ouyang, H., and Li S. (2015). A simplified binary harmony search algorithm for large scale 01 knapsack problems. *Expert Systems with Applications*, 42(12), 5337–5355.

Knowles, J.D, Thiele, L., and Zitzler, E. (2005) A tutorial on the performance assessment of stochastic multiobjective optimizers. Technical Report 214, Computer Engineering and Networks Laboratory, ETH Zurich.

Li, F., Liu, J., Tan, S., and Yu, X. (2015). R2–MOPSO: A multi-objective particle swarm optimizer based on R2-indicator and decomposition. In Proceedings of *IEEE Congress on Evolutionary Computation (CEC 2015)*, pp. 3148–3155.

Lust T., and Teghem, J. (2012). The multiobjective multidimensional knapsack problem: a survey and a new approach. *International Transactions in Operational Research*, 19(4), 495–520.

Martello, S., and Toth, T. (1990). *Knapsack Problems: Algorithms and Computer Implementations*. John Wiley & Sons, Inc.

Paquete, L,. Chiarandini, M., and Stützle, T. (2004). Pareto local optimum sets in the biobjective traveling salesman problem: An experimental study. In Xavier Gandibleux, Marc Sevaux, Kenneth Srensen, and Vincent Tkindt, editors, *Metaheuristics for Multiobjective Optimisation*, volume 535 of Lecture Notes in Economics and Mathematical Systems, pp. 177–199. Springer.

Phan, H.D., and Suzuki, J. (2013). R2-IBEA: R2 indicator based evolutionary algorithm for multiobjective optimization. In *Congress on Evolutionary Computation*, pp. 1836–1845.

Pisinger, D. (2007). The quadratic knapsack problem–a survey. *Discrete Applied Mathematics*, 155(5), 623–648.

Puchinger, J., Raidl, J.R., and Pferschy, U. (2010). The multidimensional knapsack problem: Structure and algorithms. *INFORMS Journal on Computing*, 22(2), 250–265.

Ruiz, A.B, Luque, M., Ruiz, F., and Saborido, R. (2015). A combined interactive procedure using preference-based evolutionary multiobjective optimization. Application to the efficiency improvement of the auxiliary services of power plants. *Expert Systems with Applications*, 42(21), 7466–7482.

Shojaei, H., Basten, T., Geilen, M., and Davoodi, A. (2013). A fast and scalable multidimensional multiple-choice knapsack heuristic. *ACM Transactions on Design Automation of Electronic Systems*, 18(4), 51:1–51:32.

Sinha, P. and Zoltners, A.A. (1979). The multiple-choice knapsack problem. *Operations Research*, 27(3), 503–515.

Tangpattanakul, P. (2015). Genetic algorithm and local search comparison for solving bi-objective p-Median problem. In *2015 International Conference on Informatics, Electronics Vision (ICIEV)*, pp. 1–5.

Tangpattanakul, P., Jozefowiez, N., and Lopez, P. (2015). A multi-objective local search heuristic for scheduling Earth observations taken by an agile satellite. *European Journal of Operational Research*, 245(2), 542–554.

The ministry of social affairs and health. (2012). Le champ social et médico-social: une activité en forte croissance, des métiers qui se développent et se diversifient. *Repères et Analyses*, 44.

Trautmann, H., Wagner, T., and Brockho. D. (2013). R2-EMOA: Focused multiobjective search using r2-indicator-based selection. In *7th International Learning and Intelligent Optimization Conference (LION)*, pp. 70–74. Springer.

Vianna, D.S, and Vianna, M.D. (2013). Local search-based heuristics for the multiobjective multidimensional knapsack problem. *Production*, 23(3), 478–487.

Vianna, D.S., Montané, F.A.T., Meza, E.B.M., and Martins, C.B. (2014). A memory-based GRASP heuristic for the multiobjective multidimensional knapsack problem. *International Journal of Latest Research in Science and Technology*, 3(4), 186–194.

Wagner, T., Trautmann, H., and Brockhoff, D. (2013). Preference articulation by means of the R2 indicator. In *Evolutionary Multi-Criterion Optimization*, 7811, pp. 81–95.

Wang, L., Wang, S., and Xu, Y. (2012). An effective hybrid EDA-based algorithm for solving multidimensional knapsack problem. *Expert Systems with Applications*, 39(5), 5593–5599.

Wessing, S., and Naujoks, B. (2010). Sequential parameter optimization for multi-objective problems. In *IEEE Congress on Evolutionary Computation*, pp. 1–8.

Zhang, Q., and Li, H. (2007). MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6), 712–731.

Zitzler, E., and Künzli, S. (2004). Indicator-based selection in multiobjective search. *8th International Conference on Parallel Problem Solving from Nature (PPSN VIII)*, 3242, 832–842.

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4), 257–271.

Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., and da Fonseca. V.G. (2003) Performance assessment of multiobjective optimizers: an analysis and review. *IEEE Transactions on Evolutionary Computation*, 7(2), 117–132.