Multi-criteria improvement of complex systems

Jacky Montmain, Christophe Labreuche, Abdelhak Imoussaten, François Trousset

Multi-criteria improvement of complex systems

Jacky Montmain [a,*], Christophe Labreuche [b], Abdelhak Imoussaten [a], François Trousset [a]

[a] LGI2P, EMA, Parc Scientifique Georges Besse, 30035 Nîmes, France
[b] Thales Research & Technology, 1 avenue Augustin Fresnel, 91767 Palaiseau Cedex, France

A B S T R A C T

Designing the way a complex system should evolve to better match customers' requirements provides an interesting class of applications for muticriteria techniques. The models required to support the improvement design of a complex system must include both preference models and system behavioral models. A MAUT model captures decisions related to design preferences, whereas a fuzzy representation is proposed to model relationships between system parameters and the fulfillment of system assessment criteria. The way in which these models are jointly used throughout our entire design procedure highlights that both models must be used in tandem to address managerial and implementation issues involved in an improvement project. The iterative improvement process is supported by a mathematical model, in addition to a software tool that allows our approach to be tested in an industrial case study. The search for adequate parameters regarding the improvement design is supported by a branch and bound algorithm to compute the most relevant actions to be performed. The findings confirm the efficiency of the algorithm.

Keywords:
Multiple criteria analysis
Constraint Satisfaction Problems (CSP)
Industrial performance
Management strategies
Approximate reasoning
Choquet fuzzy integral

1. Introduction

To satisfy a fluctuating demand and achieve a high level of quality and service, industries must develop and integrate new features in order to become or remain market leaders [41]. To deal with the complexity of current industrial contexts, new management strategies intended to bring about continuous improvement must take two imperatives into account: complex systems need to be tailored to an evolving context; and improvement assessment proves to be a thorny issue due to its dependence on multiple decisional aspects [4,18]. When designing improvement measures for complex systems, multiple decisions need to be considered [3]. Examples include military information architecture [47] and industrial device performance improvement [5,34]. Such settings increase the multidisciplinary design complexity regarding the fulfillment of functional, technical, environmental, economic and security requirements. In this setting, industries focus more intently on optimization and evaluation activities during the design process in order to improve and adapt complex systems. Reynoso-Meza et al. [48] explains that it is common to state a design problem as an optimization statement, where a specific cost index must be optimized. However, many real world problems require the fulfillment of a set of requirements and specifications. It is not possible to consistently transform heterogeneous factors into one single scale (*e.g.* cost) – this mindset is called *arithmo-morphism* [49]. In that case, all concerns must be taken into consideration explicitly. This is an important issue as some objectives might conflict with others, and a trade-off solution is sought.

* Corresponding author. Tel.: +33 4 66 38 70 58; fax: +33 4 66 38 70 74.
E-mail addresses: jacky.montmain@mines-ales.f (J. Montmain), christophe.labreuche@thalesgroup.com (C. Labreuche), abdelhak.imoussaten@mines-ales.f (A. Imoussaten), francois.trousset@mines-ales.f (F. Trousset).

When company designers/operators choose a new architecture to improve their system, they must first ensure that their solutions do not violate any constraints and check whether they satisfy customer needs and technical specifications, as well as the company's strategic goals and interests. Furthermore, these issues are obviously not devoid of budgetary constraints. Two extreme approaches can be adopted. In the first one, as each concern is associated with a different discipline and division within the company, the optimization regarding the different concerns can be sequentially performed after ordering the concerns according to their importance. There is no possible backtrack in this approach: the decisions made at one level (concern) are considered as requirements at the following levels. There may be conflicting requirements among the different concerns. In practice, experience is then used to try to minimize these conflicts and reach acceptable solutions [55].

The second potential extreme approach is concurrent optimization. It involves considering all concerns at the same time (concurrently) rather than considering them one at a time [28,53,38]. Then conflicting requirements, tradeoff relationships among system parameters are taken into consideration in a relevant way. This yields a single global optimization problem where the variables are the system parameter values. However, this global approach is not suited to situations where the mapping from the system parameter values to the fulfillment of strategic goals is not explicitly known, and its evaluation is very costly. Moreover, industries are more inclined to accept continuous improvement of their system rather than a more thorough perturbation of system parameters. The approach proposed in the paper is thus somewhat in-between the standard empiric sequential way (that is often conducted) and concurrent optimization. The empiric sequential approach is used in many industries for the design of complex systems. The outcomes are satisfactory (even-though sub-optimal) as designers and architects take return on past experiences and expertise into account. We propose a sequential approach, but in which backtracking is allowed. We thus propose in this work to proceed by successive improvements from an existing situation. We propose a set of possible actions on the system, where each action modifies the value of one or several system parameters.

In order to bypass the difficulty of the unknown mapping from the parameter values to the fulfillment of strategic goals, the idea is to use a behavioral model based on experience on past designs. The behavioral model of the system helps provide a simple and interpretable approximation of it, which is very useful for both operators and managers [1]. Such a model already exists in different domains: engineering (*e.g.* [29,20,19]), industry design [11], qualitative Bayesian networks [43]. As we are ultimately interested in improvements, we propose a behavioral model that relates the actions to improvements or degradations on the goals. This influence model is provided by experts and expresses their experience [37]. It has been applied to risk management and stock trading [46]. Even if two experts may theoretically come up with two different models, we do not expect large discrepancies at this stage. Apart from the influence model, two other inputs are also necessary to relate the actions to the overall impact on the satisfaction of the system. The first one synthesizes the results of the influence model. Assume that two actions are performed, the first one improves a goal and the second one is detrimental to the same goal. Then what is the overall impact on the goal? This depends on the attitude of the decision maker regarding the risk [44]. The first input describes this attitude. The second input weighs up goals and produces an overall satisfaction of the system. It is based on a multi-criteria model to aggregate the goals [52,24,25]. These two inputs are subjective and represent the decision maker's preferences.

In order to identify the set of actions that allows the decision maker to improve the overall satisfaction of the system in the most efficient way, we propose to separately deal with the multi-criteria model and the influence model (and its synthesis). The reason for this separation is that only a behavioral model of the influence of the system parameters on the fulfillment of strategic goals is known. First (at the strategic level) we start by identifying the goals for which it would be more rewarding to improve the system. Then (at the operational level) we aim at finding, through a branch-and-bound algorithm, the set of actions that would improve these goals as much as possible at the minimum cost. Although two steps are considered, backtracking is allowed when there is no set of actions that could improve the goals identified in the first step. In this case, the first step generates other goals to be improved and the optimization algorithm is launched once more.

This paper is organized as follows. Section 2 outlines a formal model for the problem of interest here. It begins by modeling the search for outputs to be improved as a multi-criteria optimization problem, before integrating this proposal into an iterative system improvement procedure. A general algorithm, based on two functions (FindCoalitions and FindActions) is proposed. Section 3 then describes function FindCoalitions: it identifies the coalition of goals/criteria to be improved first. Section 4 describes the influence model and the subjective model to synthesize its results. A branch-and-bound algorithm (function FindActions) is implemented in Section 5 as an efficient solution step. The numerical efficiency of function FindActions is analyzed in Section 6. Section 7 proposes a case study inspired from the adaptive management of a manufacturing plant. Section 8 discusses some works related to improving the competing architectures available in a multidimensional assessment context. Finally, a nomenclature of the main definitions and notations is given in Appendix A.

2. Description of the optimization algorithm

2.1. List of concepts

For starters, a complex system is characterized by input *parameters* $\gamma_1, \ldots, \gamma_p$, *e.g.* the accurate definition of all entities in a military force and its ties, or industrial device control parameters. The set of all possible parameter vector values of $(\gamma_1, \ldots, \gamma_p)$ is denoted by: $\Gamma = \Gamma_1 \times \cdots \times \Gamma_p$. A *system* is thus defined by an element $\gamma \in \Gamma$. Not all elements of Γ lead to

admissible systems for the customer since some customer requirements must generally be met. The set of elements $\Gamma_{Adm} \subseteq \Gamma$ for which the associated system satisfies these requirements yields the *feasible input parameter values*.

The company that designs or operates a complex system must therefore first satisfy the functional requirements of its customers. All elements of Γ_{Adm} satisfying customer requirements are not, however, indistinguishable from the designer's and customer's standpoint. The company has its own set of goals, priorities and strategic reasons that may favor choosing one system over another. The company then needs to construct a preference model based on criteria in line with its own goals and the customer satisfaction. Such criteria behave in a way that refines the customer's requirements within the improvement design process. The selection of a "best" solution for the company from among the set of satisfactory solutions for the customer then becomes a matter of strategic decision-making.

The set of *attributes* relative to the decision-making criteria are denoted: $X_1, \ldots, X_n$. These are the system's observable outputs. We set $N = \{1, \ldots, n\}$. Each vector of attribute values $(x_1, \ldots, x_n)$ represents a different *alternative*; the set of alternatives is then: $X = X_1 \times \cdots \times X_n$. For a military architecture, these attributes serve to quantify the way the operational mission proceeds and are obtained by large-scale simulations on systems architectures [47]. For industrial processes, these attributes might be: the work-in-progress level, flow synchronization, supplier's service rate, etc. [4]. In the industrial context, multi-criteria approaches have been, for instance, applied to flexible production lines in the manufacturing industry [44], or to portfolio selection [6].

Let $T : \Gamma_{Adm} \to X$ be the *transformation* that yields values on system attributes obtained from a vector $\gamma \in \Gamma_{Adm}$ of input parameters. We write, for $\gamma \in \Gamma_{Adm}$, $T(\gamma) = (T_1(\gamma), \ldots, T_n(\gamma))$, with $T_i : \Gamma_{Adm} \to X_i$. $T_i(\gamma)$ highlighting the impacts of the specific configuration on the attributes. On the examples given above, the transformation T cannot be precisely known in a complex system. It requires complex simulations or experiments, which are costly and time consuming. Hence transformation T generally needs to be approximated by a qualitative model. Some examples of such qualitative models are given in Sections 4.1 and 8.

The attributes must then be interpreted in terms of satisfaction with regard to the company's *goals*: a product rejection rate of 3% may be considered satisfactory for one company, whereas it remains intolerable for another, whose aim might be to achieve "*zero defect*" production in order to obtain a standard certification. Moreover, all goals are not ascribed the same relative importance in company policy. This expression of preferences tends to be complex and an elaborated multi-criteria model is required to facilitate decision-making. Among these criteria, operational and monetary criteria are typically included. From the company's standpoint, product performance is mandatory: low cost cannot compensate for poor operating performance. Consequently, the operational criteria act as a veto (a bad assessment on this criterion cannot be saved by good evaluations on the other criteria). Many other interactions among criteria, such as the conditional relative importance of criteria, are quite often encountered.

The interpretation of attributes with regard to company goals can be formalized through *utility functions*. Moreover, the relative importance of goals or their preferential interactions between one another can be handled within the Multi-Attribute Utility Theory (MAUT) framework [15,16,30]. MAUT consists of finding a real-valued utility function U such that for any pair of alternatives $x,\ x' \in X,\ U(x) \geqslant U(x')$ if and only if alternative x is at least as good as x' for the decision maker relative to all of his concerns. The most widely used model is the decomposable model developed by Krantz et al. [31], whereby U assumes the form $U(x_1, \ldots, x_n) = F(u_1(x_1), \ldots, u_n(x_n))$, with $u_i : X_i \to [0,1]$ being real-valued utility functions in $[0,1]$ and $F : [0,1]^n \to [0,1]$ an *aggregation function* [24,25]. For $x_i \in X_i,\ u_i(x_i)$ is the extent to which the goal associated with criterion i is satisfied by x_i. For $x \in X$, we set $u(x) = (u_1(x_1), \ldots, u_n(x_n))$.

A modification of input parameters from $\gamma^0 \in \Gamma_{Adm}$ to $\gamma \in \Gamma_{Adm}$ improves criterion i if $u(T_i(\gamma)) > u(T_i(\gamma^0))$. The system γ^0 is improved regarding utility function U once the new system γ satisfies $U(T(\gamma)) > U(T(\gamma^0))$.

All of these concepts involving γ, T, X, u_i, U, F are summarized in Fig. 1 (Labels L1–L5).

This formal model clearly distinguishes two points of view in the design of complex system improvements. Labels L1–L3 capture the operational viewpoint and the behavioral description of a complex system, while labels L4–L5 correspond to the strategic point of view and the preference model.

2.2. Description of the optimization problem

There are basically two high-level goals to be achieved: increased performance and minimized costs. Two alternative improvement problems can thus be derived: achieve expected overall performance at minimal cost, and maximize performance under budget constraints. Each of these optimization problems is developed in the following subsections.

2.2.1. Achieve expected overall satisfaction at minimal cost

Given the existing system characterized by parameter value $\gamma^0 \in \Gamma_{Adm}$, the problem is to find an improved solution $\gamma \in \Gamma_{Adm}$ that reaches a minimum expected overall degree of satisfaction $U_{\min}$ for $U(T(\gamma))$. Yet the company will also consider the practical cost $c(\gamma^0, \gamma)$ to improve the system γ^0 into γ. Ultimately, the company would like to determine the solution that achieves an expected overall satisfaction at the lowest cost.

$$\min\ c(\gamma^0, \gamma) \text{ under } \begin{cases} \gamma \in \Gamma_{Adm} \\ U(T(\gamma)) \geqslant U_{\min} \end{cases} \tag{1}$$

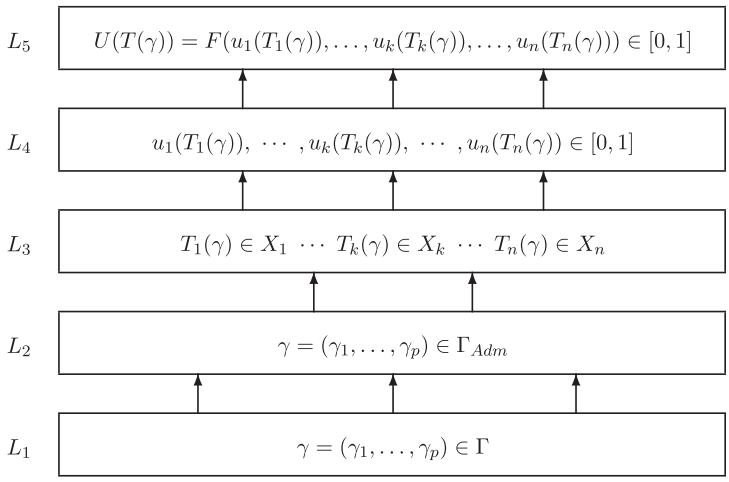

Fig. 1. Diagram of the concepts used in the evaluation of complex systems, with L1: parameter space; L2: restrictions in conjunction with the operational constraints; L3: attribute space; L4: individual satisfaction degree w.r.t. criteria; and L5: overall satisfaction. From L1 to L2: admissible systems are filtered; from L2 to L3: attributes are computed from configurations thanks to the T transformation; from L3 to L4: attributes are interpreted into satisfaction degrees; and from L4 to L5: the overall satisfaction of a configuration is computed from individual satisfaction degrees.

Solving the optimization problem stated in (1) involves performing concurrent optimization. Concurrent optimization has been successfully applied in many industrial fields [55,53,10,27,51]. For instance, genetic algorithms have been applied to find the Pareto front of optimal solutions [10]. Yet concurrent optimization has not been applied to all industrial engineering problems because concurrent engineering requires a model of the relationship between the system parameters and the goals. This corresponds to function T (going from the system parameters to the decision indicators) in Eq. (1). The main point of the paper is that there are many situations in which T is not explicitly known, and running T is very costly (it involves major simulation, or real experimentation, or changing the parameter values in a production chain – which one cannot do very often). In this case, concurrent optimization (global optimization) is not used as this would require many applications of function T, which we do not want.

In order to reduce the number of calls of function T, we propose to approximate T by a qualitative model. Such a qualitative model is widely used in engineering goal-oriented models (*e.g.* [29,20,19]), industry design [11], qualitative Bayesian networks [43]. The qualitative model considered here concerns improvements. It will be described later (see Section 4).

Using the qualitative model, we aim to perform successive improvements on the solution from the existing solution γ^0. In some domains, the user directly modifies the system by changing the parameter values γ. But in other applications, the user does not directly adjust the parameters, but he can apply some *improvement actions*. We thus introduce the set A of system improvement actions. These actions can be direct individual parameter modifications. Examples of actions having a more indirect impact on parameters are given in Section 7. Each action affects some change in some parameter(s) (*e.g.* increasing the value on a system parameter). Often several actions must be applied to be able to significantly improve the overall satisfaction.

The optimization problem formulated in (1) is thus turned into an iterative approach in which at each iteration one aims at finding the set of actions which maximizes the improvement on the overall satisfaction and minimizes the cost increase. When applying these actions, γ is changed. Then one computes $U(T(\gamma))$. The process continues until $U(T(\gamma))$ reaches the minimal expected value $U_{\min}$. This approach can be seen as a kind of steepest descent method.

The qualitative model actually directly relates each individual action to each criterion, in terms of the extent of belief that an improvement or a decrease regarding the criterion occurs. There is no direct information on the intensity of the increase/decrease that is expected. It is thus impossible to assess the impact on the overall evaluation. Hence the set of actions which maximizes the improvement of the overall satisfaction and minimizes the increase of the cost is determined in two separate steps – represented by functions "FindCoalition" and "FindActions":

- First, at the strategic level, we identify the criteria whose value can be modified to achieve the greatest reward. Given $\gamma \in \Gamma$, we first compute the vector $u(T(\gamma)) = (u_1(T_1(\gamma)), \ldots, u_n(T_n(\gamma))) \in [0,1]^N$ providing the satisfaction values on n criteria. Function FindCoalition$(\mathcal{B}, u)$ determines the coalition I^* of criteria belonging to the set $\mathcal{B}$ of acceptable coalitions, for which the improvement of $u_i(T_i(\gamma))$ on all criteria i in I^* improves the overall satisfaction in the most efficient way. We start with $\mathcal{B} = 2^N$, with 2^N being the power set of N. This takes both the aggregation function F and the cost of the improvement into account.
- Second, at the operational level, one searches for the set of actions which allows improvement of all criteria in I^* with the best expectation. Function FindActions(A, I^*) returns several alternative sets of actions to be performed. The operator is asked to select the most appropriate one.

```
Function GeneralProcess1(A, U_min, γ) :
    u ← u(T(γ)).
    U ← U(T(γ)).
    While ( U < U_min) do
        B ← 2^N.
        Repeat
            I* ← FindCoalition(B, u ).
            S ← FindActions(A, I*).
            B ← B \ {I ⊆ N : I ⊇ I*}.
        until (S ≠ ∅)
        The user selects one ap ∈ S and applies it. This modifies
        γ.
        u ← (u_1(T_1(γ)), ..., u_n(T_n(γ))).
        U ← U(T(γ)).
    done
    return γ;
End
```

Algorithm 1. General algorithm GeneralProcess$(A, U_{\min}, \gamma)$, where A is the set of actions, $U_{\min}$ is the minimal satisfaction and γ is the initial system parameter value. Initially, we call GeneralProcess$(A, U_{\min}, \gamma^0)$.

This two-step approach is described in Algorithm 1. FindCoalition and FindActions are described in Sections 3 and 5, respectively.

The main original feature of Algorithm 1 is that it provides the possibility of exploring several subsets I^* (in the Repeat-until loop), which permits backtracking. This avoids incompatibility between "FindCoalition" and "FindActions". More precisely, it might be possible that there is no set of actions that improves all criteria I^* identified at the first step. In this case, another coalition I^* has to be found. If it is not possible to improve all criteria in I^* at the same time, it will not be possible to improve all criteria in a superset of I^*. Hence all supersets of I^* are discarded from the list of admissible sets of criteria for $\mathcal{B}$.

2.2.2. Maximize satisfaction under budget constraints

The company may alternatively prefer to determine the solution that maximizes performance under budget constraints:

$$\max\ U(T(\gamma)) \text{ under } \begin{cases} \gamma \in \Gamma_{Adm} \\ c(\gamma^0, \gamma) \leqslant C_{\max} \end{cases} \tag{2}$$

Algorithm 1 can easily be modified to address the optimization problem stated in (2) (see Algorithm 2). Hereafter we will focus on the optimization problem stated in (1) and Algorithm 1.

In Algorithm 2, an operational cost $c_{op}(a)$ is associated with each action a. More precisely for a subset ap of actions, $c_{op}(ap)$ is the sum of the operational cost of each action in ap : $c_{op}(ap) = \sum_{a \in ap} c_{op}(a)$.

Algorithm 2 finds successive actions to be performed, while keeping the improvement cost under $C_{\max}$. Note that condition $c + c_{op}(ap) \leqslant C_{\max}$ is automatically satisfied for every $ap \in \text{FindActions}(A, I^*)$ if this function is called with $budget = C_{\max} - c$ (see Section 5).

3. Function FindCoalition: improvement recommendation at the strategic level

The MAUT framework merely deals with designers' preferences without any further considerations regarding the material constraints beyond improvement implementation. Given a solution described by vector $u \in [0,1]^n$ of scores w.r.t. criteria satisfaction degrees, the function FindCoalition$(\mathcal{B}, u)$ returns the set I^* of criteria for which it is the most rewarding to improve the solution, independently of any operational constraints. The set I^* results from a benefit to cost ratio analysis. On the one hand, as I^* increases the benefit increases thanks to aggregation function F. On the other hand, as I^* increases the cost to perform an improvement increases for all criteria in I^*. The difficulty here is that the intensity of the improvement actions on criteria is unknown. Hence for each subset $I \subseteq N$, an indicator that assesses the average benefit to cost ratio of improving criteria in I will be defined. Then I^* is defined as the subset I for which the associated indicator is maximal. Let us first consider when there is no improvement cost.

An index (called worth index) denoted $\omega_I(F)(u)$ and quantifying the worth of improving vector u in criteria among $I \subseteq N$, and subject to the evaluation function F, has been proposed in [33]. As exhibited in the following example, subsets I should not be restricted to single entries. Let us consider the case of a highly intolerant designer, as described by the *min* aggregation function: $F(u) = \min_{i \in N} u_i$. Should all the criteria be equally satisfied, then improving just one criterion will not alter the overall evaluation. Hence, it is fruitless to work on a single criterion and instead it is worth improving all criteria simultaneously.

Function GeneralProcess2($A, C_{\max}, \gamma$) :
  $u \leftarrow u(T(\gamma))$.
  $c \leftarrow 0$.
  Repeat
    $\mathcal{B} \leftarrow 2^N$.
    Repeat
      $I^* \leftarrow$ FindCoalition($\mathcal{B}$, u).
      $\mathcal{S} \leftarrow \{ap \in$ FindActions(A, I^*) s.t. $c + c_{\text{op}}(ap) \leq C_{\max}\}$.
      $\mathcal{B} \leftarrow \mathcal{B} \setminus \{I \subseteq N : I \supseteq I^*\}$.
    until ($\mathcal{B} = \emptyset$ **OR** $\mathcal{S} \neq \emptyset$)
    If ($\mathcal{S} \neq \emptyset$) **then**
      The user selects one $ap \in \mathcal{S}$ and applies it. This modifies γ.
      $u \leftarrow u(T(\gamma))$.
      $c \leftarrow c + c_{\text{op}}(ap)$.
    end If
  until ($\mathcal{S} \neq \emptyset$)
  return γ;
End

Algorithm 2. Variant of the general algorithm, for the optimization problem formulated in (2).

Let V be the set of piecewise continuous functions defined on $[0,1]^n$. This space is endowed with the norm $\forall H \in V, \|H\|_V = \sup_{x\in[0,1]^n} |H(x)|$. The index ω_I is considered as an operator from V into itself. The index ω_I is defined axiomatically for any $F \in V$. First, if F is constant over criteria in I, then $\omega_I(F)(u) = 0$. Moreover, if F is independent of criterion i, then $\omega_{I\cup\{i\}}(F)(u) = \omega_I(F)(u)$. Another requirement whenever F can be decomposed into n functions F_i for each criterion, is to provide a description of an optimistic decomposability of $\omega_I(F)$ from the $\omega_i(F_i)$. Lastly, an invariance property $\omega_I(F)$ for $\{0,1\}$-valued functions F is described. Previous requirements combined with linearity, symmetry and continuity (*i.e.* $\sup_{F\in V, F\neq 0} \frac{\|\omega_I(F)\|_V}{\|F\|_V} < \infty$) serve to uniquely define ω_I [33]:

$$\omega_I(F)(u) = \int_0^1 \left[F((1-\tau)u_I + \tau 1_I, u_{N\setminus I}) - F(u)\right] d\tau \tag{3}$$

where 1_I is the identity function. For every $d \in [0,1]^n$, $(d_I, u_{N\setminus I})$ denotes a performance vector, with a component of d on criteria in I, and the components of u on the other criteria. With the notation $d_I = (1-\tau)u_I + \tau 1_I$, the right hand side of Eq. (3) gives the mean value of the gain $F(d_I, u_{N\setminus I}) - F(u)$ only for improvement vectors d_I on the segment from the current performance values u_I (for $\tau = 0$) to the best possible improvement in I, *i.e.* 1_I (for $\tau = 1$). Eq. (3) therefore yields the mean impact generated by uniformly improving all criteria in I simultaneously, at which point it may be assumed that all possible levels of improvement (from sticking to u_I until reaching the ideal vector 1_I) have the same probability of occurring. The diagonal from u_I to 1_I is considered since it has been assumed that the improvements on all criteria are homogeneous.

Let us now introduce the cost in Eq. (3). The *cost at the strategic level* going from satisfaction profile u to u' is denoted $c_{\text{str}}(u, u')$.

Note that this cost may be correlated with decisive factors that do not necessarily entail monetary considerations, they might actually be correlated with risk appraisal, temporal requirements, resources availability, etc. In such cases, no relationship whatsoever exists between cost functions $c_{\text{str}}(u, u')$ and $c(\gamma, \gamma')$, and $c_{\text{str}}(u, u')$ ensures an additional degree of freedom in the improvement design.

Eq. (3) can be generalized by replacing the benefit factor by a benefit to cost ratio:

$$\omega_I(F)(u) = \int_0^1 \frac{\left[F((1-\tau)u_I + \tau 1_I, u_{N\setminus I}) - F(u)\right]}{c_{\text{str}}(u, ((1-\tau)u_I + \tau 1_I, u_{N\setminus I}))} d\tau \tag{4}$$

Under the linear assumption $c_{\text{str}}(u, u') = \sum_{i\in N}(u'_i - u_i)uc^i_{\text{str}}$, with uc^i_{str} being a unit cost related to criterion i. Hence $c_{\text{str}}(u, ((1-\tau)u_I + \tau 1_I, u_{N\setminus I})) = \tau\sum_{i\in I}(1-u_i)uc^i_{\text{str}}$. When $c_{\text{str}}(u, u')$ is related to monetary considerations, value $c_{\text{str}}(u, ((1-\tau)u_I + \tau 1_I, u_{N\setminus I}))$ will be considered as a mean cost for improving u into $((1-\tau)u_I + \tau 1_I, u_{N\setminus I})$, which is assessed independently of any particular improvement action. This value can only be produced when the designer's/operator's know-how is available. For example, an experienced designer/operator is assumed to be capable of assessing the cost related to employee training as a yearly average without precise knowledge of the allocated training actions.

Finally, the subset I^* of criteria that maximizes the worth index

$$\text{FindCoalition}(\mathcal{B}, u) = \text{Arg} \max_{I \in \mathcal{B}} \omega_I(F)(u) \tag{5}$$

indicates the mean satisfaction to cost ratios that it would be most beneficial to improve first.

4. Concepts for improvement recommendations at the operational level

The operational constraints were ignored during the identification of I^*. In function FindActions(A, I^*), these operational constraints are considered, one aims at finding how the designer/operator should modify system γ in order to improve criteria in I^*.

Before defining FindActions(A, I^*), the qualitative influence model has to be specified.

4.1. Goals-actions relationships

According to our approach, system improvements are carried out through system improvement actions. The set of actions is denoted A. In order to improve a subset of criteria, several actions generally need to be performed simultaneously. As there are some constraints among the actions, some of them cannot be performed together: they are said to be *mutually exclusive*. An action plan is a set of actions that can be performed together. Let us begin by specifying the action plan concept.

Definition 1. An *action plan ap* is a subset of non-exclusive actions, *i.e.* $ap \subseteq A$. The set of all action plans is denoted $AP \subseteq 2^A$.

In general, the set of actions and action plans can be defined independently of the system parameters. However, actions can be defined directly as modifications of the system parameters.

Example 2. Let us consider the case when two actions are related to each parameter γ_j: this value can be increased or decreased. The two actions "γ_j increase" and "γ_j decrease" are mutually exclusive. Thus, $2p$ potential actions are available in A. Action plans are subsets of actions that concern different parameters.

We are interested in system improvements. Hence, instead of defining the complex transformation $T : \Gamma_{Adm} \to X$ that yields the values on system attributes obtained from a vector $\gamma \in \Gamma_{Adm}$, experts are asked to provide the relationship between actions in A and the goals (criteria) in a qualitative way, as is often the case in goal oriented models (*e.g.* [18,20,19]). We will use a fuzzy behavioral model. Indeed, as the gathered information originates from the experts' or managers' perception rather than being factually measured, it is intrinsically imprecise [54,37].

Our proposal is based on the pioneering work of [11]. Felix's fuzzy relationship model introduces two fuzzy subsets to distinguish actions with positive or negative impacts on performance [13].

Definition 3. Consider a criterion $i \in N$ and an action $a \in A$. Let $S_i(a)$ be the degree of belief to which action a can positively affect criterion i. When $S_i(a) > 0$, we say that action a *satisfies* i. Let $D_i(a)$ be the degree of belief to which action a can negatively affect criterion i. When $D_i(a) > 0$, we say that action a *distracts* i. In other words S_i (resp. D_i) can be seen as the fuzzy subset of actions which support (resp. distract) i. Action a has no influence on i when $S_i(a) = 0$ and $D_i(a) = 0$. The influence of an action is either positive, negative or nul: $\min(S_i(a), D_i(a)) = 0$.

By considering the influence of actions over criteria directly without explicitly going through the system parameters, we do not need to take constraints among parameter values into account.

From our perspective, this fuzzy model would seem to match the genuine expertise [1] generally available as a result of transforming $u \circ T$ into a complex system: the impact of an action on system performance can typically only be described in a qualitative and non-deterministic manner.

The fuzzy model described in Definition 3 can be represented through a digraph between A and N, such that (see example in Fig. 2): the arc between action a and criterion i is equal to

$$\text{Influence}(a, i) = \begin{cases} +S_i(a) & \text{if } S_i(a) > 0 \\ -D_i(a) & \text{if } D_i(a) > 0 \end{cases} \tag{6}$$

4.2. Combining the influences of several actions: reasoning framework

The qualitative action–criteria relationship needs to be extended to action plans. The major difficulty is that for an action plan ap and a criterion i, several actions in ap may affect i positively and several other actions in ap may affect i negatively. Then what is the resulting effect of ap on criterion i?

This problem could be solved by asking the experts to provide the degree of belief to which an action plan is supported to directly affect a criterion. This is not realistic in practice due to the huge size of AP. The estimation of the merged impact of an action plan naturally depends on the system behavior, as well as on the designer's/operator's decisional behavior: a pessimistic attitude (whereby a risk aversion position will focus attention on the most highly negative merged impacts of the

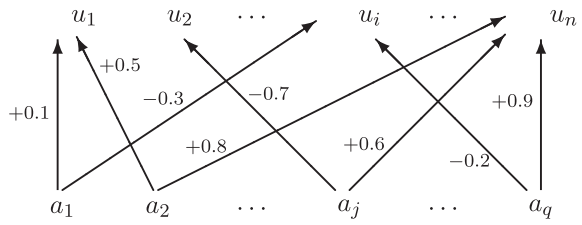

Fig. 2. Digraph of action–criteria relationships.

action plan) vs. an optimistic attitude (whereby risk acceptance will focus attention on the most highly positive merged impacts). Hence the combination of positive and negative impacts cannot be solved objectively and is equivalent to a problem of decision under uncertainty.

Consider an action plan $ap \in AP$ and a criterion $i \in N$. We wish to define the resulting effect of ap over i. This will be based on actions $A_i^S(ap)$ (resp. $A_i^D(ap)$) that have a positive (resp. negative) effect on i:

$$A_i^S(ap) = \{a \in ap : S_i(a) > 0\} \quad \text{and} \quad A_i^D(ap) = \{a \in ap : D_i(a) > 0\} \tag{7}$$

The influence is basically bipolar as we can distinguish between positive (support) and negative (denial) influences. We do not wish to directly synthesize these contradictory stimuli. Hence we are in the *bivariate context*, where the resulting effect of ap over i is first assessed using two separate scales [22]: one for positive stimuli in favor of supporting i, and one for negative stimuli in favor of denial.

Definition 4. The bivariate scale is denoted by the pair $(S_i(ap), D_i(ap))$, where $S_i(ap)$ (resp. $D_i(ap)$) is the degree to which ap satisfies (resp. denies) i.

This bivariate scale is for instance used in [19]. $S_i(ap)$ (resp. $D_i(ap)$) results from an aggregation of numbers $\{S_i(a), a \in A_i^S(ap)\}$ (resp. $\{D_i(a), a \in A_i^D(ap)\}$). The choice of aggregation functions for the satisfaction/denial parts depends on the attitude of the designer/operator regarding risk.

Definition 5. Let us give two standard risk behaviors:

- Under *Drastic Reasoning Framework* (*DRF*), we have

  $$S_i^{\text{DRF}}(ap) = \min_{a \in A_i^S(ap)} S_i(a) \quad \text{and} \quad D_i^{\text{DRF}}(ap) = \max_{a \in A_i^D(ap)} D_i(a)$$

  According to *DRF*, the lowest belief in a positive impact value is to be compared to the highest belief in a negative impact value on i. The designer's/operator's behavior represents a serious risk aversion and also indicates a pessimistic attitude.
- Under *Flexible Reasoning Framework* (*FRF*), we have

  $$S_i^{\text{FRF}}(ap) = \max_{a \in A_i^S(ap)} S_i(a) \quad \text{and} \quad D_i^{\text{FRF}}(ap) = \max_{a \in A_i^D(ap)} D_i(a)$$

  According to *FRF*, the highest belief in a positive impact value is to be compared to the highest belief in a negative impact value on i. In this instance, the designer's/operator's behavior more willingly accepts being exposed to the risk of committing an estimation error. This framework describes an optimistic attitude.

Other aggregation functions are of course possible, but we will stick to the previous two examples as they represent the most standard forms. Note that *FRF* is chosen in [19].

We now need to quantify the degree of belief to which an action plan is expected to improve a subset of criteria in I^*.

Definition 6. Consider an action plan $ap \in AP$ and a criterion i. The resulting degree of belief to which ap should contribute to the improvement of i is:

$$s_i(ap) = \begin{cases} S_i(ap) & \text{if } S_i(ap) > D_i(ap) \\ 0 & \text{else} \end{cases} \tag{8}$$

Note that condition "$S_i(ap) > D_i(ap)$" can be replaced by a more drastic condition such as "$S_i(ap) > D_i(ap) + \eta$" (where $\eta > 0$ is a threshold). Definition 6 concerns a single criterion but can be extended to any subset of criteria. We will define the degree of belief $s_I(ap)$ to which ap should improve all criteria in I. Two proposals are put forward.

Definition 7. Consider a subset of criteria $I \subseteq N$ and an action plan $ap \in AP$. The resulting degree of belief to which ap should contribute to the improvement of I is

$$\tilde{s}_I(ap) = \min_{i \in I} s_i(ap) \tag{9}$$

The previous definition can be enhanced, by requiring no degradation on the remaining criteria $N \setminus I$.

Definition 8. Consider a subset of criteria $I \subseteq N$ and an action plan $ap \in AP$. The resulting degree of belief to which ap should contribute to the improvement of I while not deteriorating the other criteria is

$$\hat{s}_I(ap) = \begin{cases} \min\limits_{i \in I} s_i(ap) & \text{if } \forall j \in N \setminus I,\ [S_j(ap) > D_j(ap) \text{ or } A_j^D(ap) = \emptyset] \\ 0 & \text{else} \end{cases} \tag{10}$$

In other words, $\hat{s}_I(ap) > 0$ if any criterion in I is improved by ap, whereas no criterion stated in $N \setminus I$ is deterred. For criterion j in $N \setminus I$, note that the condition in (10) is fulfilled when $A_j^D(ap) = A_j^S(ap) = \emptyset$.

Note nevertheless that when the vector of criteria satisfaction degrees is Pareto optimal, improving criteria in I will necessarily give rise to a negative impact with regard to certain other criteria, and then the constraint in equation stated in (10) cannot be satisfied. For example, in the *DRF* framework, a slight negative impact can be tolerated for criteria in a subset $I' \subseteq (N \setminus I)$ and then constraint $[S_j(ap) > D_j(ap)$ or $A_j^D(ap) = \emptyset]$ is maintained in Eq. (10) only for criteria in $N \setminus (I \cup I')$.

Hereafter, $s_I(ap)$ will be equal either to $\hat{s}_I(ap)$ or to $\tilde{s}_I(ap)$.

Definition 9. An action plan $ap \in AP$ is admissible on I^* if $s_{I^*}(ap) > 0$. ap is admissible with degree $\alpha \in]0, 1]$ if $s_{I^*}(ap) \geqslant \alpha$.

4.3. Aim of function FindActions(A, I^*)

Designers/operators focus on solutions with degrees of admissibility at least as high as a threshold α_0 and whose cost does not exceed the budget constraint (*budget* threshold). Nevertheless, they would naturally prefer the solution with the highest degree of admissibility at an identical cost; conversely, they would prefer the least costly solutions should the degree of admissibility be the same. An order of preference on action plans is defined in order to represent this behavior. To develop this order, let us consider the non-dominated Pareto action plans with respect to these two metrics: degree of admissibility $s_{I^*}(ap)$ and operational cost $c_{\text{op}}(ap)$, c_{op} is defined in Section 2.2.2. Since the preferences governing these two metrics lie in opposite directions, we apply the Pareto order with respect to the couple $(s_{I^*}(ap), -c_{\text{op}}(ap))$. We make the following assumptions

$$\alpha_0 > 0, \quad budget > 0, \quad \forall a \in A \quad c_{\text{op}}(a) > 0 \tag{11}$$

Let us define the Pareto order $\prec$ over action plans admissible of AP as:

$$\begin{aligned} ap \prec ap' \iff & (s_{I^*}(ap), -c_{\text{op}}(ap)) \prec_{\text{Pareto}} (s_{I^*}(ap'), -c_{\text{op}}(ap')) \iff [(s_{I^*}(ap) < s_{I^*}(ap')) \text{ and } (c_{\text{op}}(ap) \\ & \geqslant c_{\text{op}}(ap'))] \text{ or } [(s_{I^*}(ap) \leqslant s_{I^*}(ap')) \text{ and } (c_{\text{op}}(ap) > c_{\text{op}}(ap'))] \end{aligned} \tag{12}$$

Action plan ap is dominated by action plan ap' if $ap \prec ap'$. The search for efficient admissible action plans, with a cost lower than *budget* and admissibility greater than α_0, can then be stated as:

$$\max_{ap \in AP : s_{I^*}(ap) \geqslant \alpha_0 \text{ and } c_{\text{op}}(ap) \leqslant budget} (s_{I^*}(ap), -c_{\text{op}}(ap)) \tag{13}$$

5. Function FindActions: improvement recommendation at the operational level

The objective of this section is to determine a set of admissible action plans at minimum cost and maximum degree of admissibility. In practice, the Pareto front of solutions for the problem stated in Eq. (13) is computed. This section will also present a branch-and-bound algorithm with appropriate heuristics in order to efficiently solve this problem [8]. These heuristics depend on the adopted reasoning framework (*DRF* or *FRF* – see Definition 5).

Furthermore, several framework-dependent definitions are to be introduced in order to help explain the algorithm heuristics.

Definition 10. Let $i \in N$

- *improvement*: for $i \in N \setminus I^*$, action plan ap *improves* criterion i when $A_i^D(ap) = \emptyset$ or $S_i(ap) > D_i(ap)$. For $i \in I^*$, ap *improves* i when $S_i(ap) > D_i(ap)$,
- *hindrance*: action plan ap *hinders* criterion i if $A_i^D(ap) \neq \emptyset$ and $S_i(ap) \leqslant D_i(ap)$,
- *compensation*: action a *compensates* for hindering actions in action plan ap on criterion i if $S_i(a) > \max_{a' \in A_i^D(ap)} D_i(a')$,
- *restriction*: an action a is *restrictive* if $D_i(a) < \alpha_0$ for $i \in I^*$ and in case *FRF*, a does not compensate for any hindering actions for performances in $N \setminus I^*$. An action a is also *restrictive* if $c_{op}(a) > budget$.

5.1. The general branch and bound principle

The branch and bound algorithm explores the set AP of action plans. We assume here that there is no constraint among the individual actions such that $AP = 2^A$. Relaxing this assumption would require the use of Constraint Solving Problem (CSP) techniques. We decided to not consider this option because it would considerably lengthen the paper.

Introducing heuristics into the branch-and-bound algorithm allows selection of a relevant representation to efficiently derive solutions while cutting irrelevant branches as quickly as possible.

The general branch-and-bound algorithm (SOLVE) (see Algorithm 3) computes efficient admissible action plans by constructing and returning the Pareto front of solutions (Eq. (13)). The SOLVE function parameters are: B (set of remaining actions, *i.e.* potential candidate actions not yet included in the action plan); I^* the set of criteria to be first improved; ap (the action plan under construction, such that $ap \cap B = \emptyset$); and $\mathcal{S}$ (the Pareto front of previously known solutions). It is initially launched as FindActions$(A, I^*) =$ SOLVE$(A, I^*, \emptyset, \emptyset)$.

According to Algorithm 3, the few lines just after ① update the Pareto front of solutions when ap is added. IS_ADMISSIBLE basically computes $(s_{I^*}(ap) \geqslant \alpha_0) \wedge (c_{\text{op}}(ap) \leqslant budget)$. In the algorithm, we also add the fact that *FRF* is used to this condition. Indeed, even if ap is admissible its admissibility degree can be increased by adding some actions to ap under *FRF*. CHOOSE introduces heuristics to select and return the action to be chosen at the current node (Section 5.2). Lastly, REDUCE introduces heuristics to cut branches as quickly as possible by reducing the set of remaining candidate actions (Section 5.3).

5.2. Action selection (CHOOSE)

The purpose of action selection CHOOSE(B, ap) is to enhance the search by choosing a relevant candidate action capable of leading to a conclusion (whether positive or negative) as quickly as possible. Heuristics are then used to select such an action with minimal computations. In *DRF*, the action selected is the one that generates the most drastic constraints to enable branch-cutting as quickly as possible. In *FRF*, supplementary heuristics related to the model specificities are added. An action selected with these heuristics hinders at least one criterion in N, with as few as possible remaining actions capable of compensating for this hindrance. In both cases, the selected actions will severely constrain the next selections when added to ap, hence reducing the search complexity of this branch. More precisely, two sets are introduced.

Definition 11. For $a \in B$, set $DIS(a, ap, B) = \{i \in N : a \in A_i^D(B) \text{ and } D_i(a) > \max(S_i(ap), D_i(ap))\}$, where $A_i^D(B)$ is defined in Eq. (7).

$DIS(a, ap, B)$ is the set of criteria such that a acts negatively on i ($a \in A_i^D(B)$) with a degree greater than that involved in $S_i(ap)$ and $D_i(ap)$.

```
Function SOLVE( B, I*, ap, S ) : → Pareto-Front-of-Solutions
    S' ← S;
    If ( IS_ADMISSIBLE(ap, I*) ) then
        // ①: ap is admissible – Add ap to S if non-dominated
        If ( ∄ap' ∈ S / ap ≺ ap' ) then
            S' ← {ap} ∪ {ap' ∈ S / ap' ⊀ ap};
        end If
    end If
    If ( ¬IS_ADMISSIBLE(ap, I*) ∨ framework uses FRF ) then
        // ②: Remove from B the elements that yield non-admissibility
        B' ← REDUCE ( B , ap );
        If ( B' ≠ ∅ ) then
            // ③: Select one element in B'
            a ← CHOOSE ( B' , ap );
            // ④: Explore the selection of the other elements of B'
            S'' ← SOLVE ( B' \ {a} , I* , ap , S' );
            // ⑤: Explore when a is added to ap
            S' ← SOLVE ( B' \ {a} , I* , ap ∪ {a} , S'' );
        end If
    end If
    return S';
End
```

Algorithm 3. Determination of the Pareto front of solutions $\mathcal{S}$.

Lemma 12. *For every $i \in DIS(a, ap, B)$, condition $S_i(ap \cup \{a\}) > D_i(ap \cup \{a\})$ does not hold.*

Proof. Let $i \in DIS(a, ap, B)$ and denote $ap' = ap \cup \{a\}$. Firstly, as $a \in A_i^D(B)$, then $A_i^D(ap') \neq \emptyset$ (note that $a \in A_i^D(ap')$). Secondly, we have in both frameworks *DRF* and *FRF*: $S_i(ap') = S_i(ap) < D_i(a) = \max_{b \in ap'} D_i(b) = D_i(ap')$. Hence $S_i(ap') > D_i(ap')$ does not hold. □

Lemma 13. *Let $\beta \in]0, 1], i \in N$, and set $IMP(i, \beta, B, ap) = \{a \in B : a \in A_i^S(B), S_i(a) \geqslant \beta$ and $S_i(ap) < \beta\}$. Let $a' \in B$ such that $a' \in A_i^D(B)$, actions in $IMP(i, D_i(a'), B, ap)$ can compensate for hindering action a' when ap cannot guarantee the compensation.*

Proof. Obvious. □

The two previous sets in Lemmas 12 and 13 allow us to define, for B and ap:

$$Co(B, ap) = \{a \in B : \exists i \in DIS(a, ap, B) \text{ s.t. } IMP(i, D_i(a), B, ap) \neq \emptyset\}$$

and (where *MCo* stands for Most Constraining selection)

$$MCo(B, ap) = \operatorname*{argmin}_{a \in B} \min_{i \in DIS(a, ap, B)} |IMP(i, D_i(a), B, ap)|$$

Finally, CHOOSE(B, ap) returns any element in $MCo(B, ap)$.

5.3. Reducing the set of remaining actions (REDUCE)

REDUCE aims to remove ineffective actions from the remaining set of actions. Each time an action is deleted, the search complexity of the particular branch is halved. REDUCE is based on several iteratively applied elementary reductions that are mutually exclusive, non-compensable, expensive, Pareto, or else related to the cardinal of ap. Some reductions may depend on the choice of reasoning framework among *DRF* or *FRF* and of synthesis function among $\tilde{s}$ and $\hat{s}$. To avoid lengthening this paper, we only present five elementary reductions.

5.3.1. Cardinality reduction

Lemma 14. *Let ap be an action plan improving criteria in I^*. If $|ap| > |N|$, then there exists $ap' \subseteq ap$ with $|ap'| \leqslant |N|$ such that $ap \prec ap'$.*

Proof. For every $i \in N$, the action $a \in ap$ that achieves $\max_{a \in ap} S_i(a)$ is denoted $a(i)$.

Let us define J as follows: when $\hat{s}$ is used, set $J = \{i \in N \setminus I^* : A_i^S(ap) \neq \emptyset\}$; and when $\tilde{s}$ is used, set $J = \emptyset$. Let $ap' = \{a(i) : i \in I^* \cup J\}$. Clearly $|ap'| \leqslant |N|$ and thus $c_{\text{op}}(ap') < c_{\text{op}}(ap)$.

As ap is then admissible for every $i \in I^* \cup J$, every $a \in A_i^S(ap)$, and for propagation models *DRF* and *FRF*:

- For *DRF*:

$$\min_{a \in A_i^S(ap')} S_i(a) \geqslant \min_{a \in A_i^S(ap)} S_i(a) > \max_{a \in A_i^D(ap)} D_i(a) \geqslant \max_{a \in A_i^D(ap')} D_i(a)$$

- For *FRF*:

$$\max_{a \in A_i^S(ap')} S_i(a) = \max_{a \in A_i^S(ap)} S_i(a) > \max_{a \in A_i^D(ap)} D_i(a) \geqslant \max_{a \in A_i^D(ap')} D_i(a)$$

When $\hat{s}$ is used, if $i \in N \setminus (I^* \cup J)$, we have $A_i^D(ap) = \emptyset$ since ap is admissible and $A_i^D(ap') = \emptyset$. Hence $s_{I^*}(ap') \geqslant s_{I^*}(ap)$ in propagation models *DRF* and *FRF*. □

This implies that branches with more than $|N|$ elements are cut.

5.3.2. Locking Action reduction (LA)

Let us define $LA(B, ap)$ as follows: when $\tilde{s}$ is used, we set

$$LA(B, ap) = \left\{a \in B : \exists i \in I^* \text{ s.t. } D_i(a) \geqslant \max_{a' \in B \cup ap} S_i(a')\right\}$$

and when $\hat{s}$ is used, we set

$$LA(B, ap) = \left\{a \in B : \exists i \in N \text{ s.t. } D_i(a) \geqslant \max_{a' \in B \cup ap} S_i(a')\right\}$$

Each locking action in $LA(B, ap)$ would distract from criteria in N which could be compensated for by any action in B or ap. Indeed any action plan containing such an action will not be an admissible action plan due to the relations stated in formulas (9) and (10). Hence the following result is shown.

Lemma 15. *There does not exist any $ap' \subseteq B \cup ap$ with $ap \subseteq ap'$ and $LA(B, ap) \cap ap' \neq \emptyset$ which is admissible.*

5.3.3. Cost reduction (LC)

Let $LC(B, ap) = \{a \in B : c_{\text{op}}(ap \cup \{a\}) > budget\}$. None of the actions in $LC(B, ap)$ can be added to ap without exceeding the maximal allowed cost *budget*. Hence the following result is shown.

Lemma 16. *There does not exist any $ap' \subseteq B \cup ap$ with $ap \subseteq ap'$ and $LC(B, ap) \cap ap' \neq \emptyset$ which is admissible.*

5.3.4. Incompatible Admissibility Reduction (INC)

This reduction only applies in framework *DRF*. Let $INC(B, ap) = \{a \in B : \exists i \in N, \exists a' \in ap \text{ s.t. } S_i(a') \leqslant D_i(a) \text{ or } D_i(a') \geqslant S_i(a)\}$.

Lemma 17. *There does not exist any $ap' \subseteq B \cup ap$ with $ap \subseteq ap'$ and $INC(B, ap) \cap ap' \neq \emptyset$ which is admissible.*

Proof. Adding $a \in INC(B, ap)$ to ap implies that there exists $i \in I^*$ such that

- in the case where there exists $a' \in ap$ s.t. $S_i(a') \leqslant D_i(a)$, then $S_i^{\text{DRF}}(ap \cap \{a\}) \leqslant S_i(a') \leqslant D_i(a) \leqslant D_i^{\text{DRF}}(ap \cap \{a\})$;
- in the case where there exists $a' \in ap$ s.t. $D_i(a') \geqslant S_i(a)$, then $S_i^{\text{DRF}}(ap \cap \{a\}) \leqslant S_i(a) \leqslant D_i(a') \leqslant D_i^{\text{DRF}}(ap \cap \{a\})$.

In both cases, $\tilde{s}_{I^*}(ap \cap \{a\}) = 0$ and $\hat{s}_{I^*}(ap \cap \{a\}) = 0$. Under *DRF*, adding other actions cannot improve s_{I^*}. □

5.3.5. Pareto Reduction (PAR)

Under *DRF*, we define $PAR_{DRF}(B, ap, \mathcal{S})$ by

$$\left\{ ap' \in \mathcal{S} : \left[\left(\min_{\{i \in I^* / A_i^{\mathcal{S}}(ap) \neq \emptyset\}} S_i^{DRF}(ap) < s_{I^*}(ap') \right) \wedge \left(c_{op}(ap) \geqslant c_{op}(ap') \right) \right] \vee \left[\left(\min_{\{i \in I^* / A_i^{\mathcal{S}}(ap) \neq \emptyset\}} S_i^{DRF}(ap) = s_{I^*}(ap') \right) \wedge \left(c_{op}(ap) > c_{op}(ap') \right) \right] \right\}$$

Lemma 18. *If $PAR_{DRF}(B, ap, \mathcal{S}) \neq \emptyset$ then for all $ap'' \subseteq B \cup ap$ where $ap \subset ap''$ and ap'' is admissible, there exists $ap' \in \mathcal{S}$ such that $ap'' \prec ap'$.*

Proof. Let $ap'' \subseteq B \cup ap$ where $ap \subset ap''$ and ap'' is admissible. As $PAR_{DRF}(B, ap, \mathcal{S}) \neq \emptyset$, there exists $ap' \in \mathcal{S}$ such that:

- $(s_{I^*}(ap'') \leqslant \min_{\{i \in I^* / A_i^{\mathcal{S}}(ap) \neq \emptyset\}} S_i^{DRF}(ap) < s_{I^*}(ap')) \wedge (c_{op}(ap'') > c_{op}(ap) \geqslant c_{op}(ap'))$ then $(s_{I^*}(ap'') < s_{I^*}(ap')) \wedge (c_{op}(ap'') > c_{op}(ap'))$
- Or $(s_{I^*}(ap'') \leqslant \min_{\{i \in I^* / A_i^{\mathcal{S}}(ap) \neq \emptyset\}} S_i^{DRF}(ap) = s_{I^*}(ap')) \wedge (c_{op}(ap'') > c_{op}(ap) > c_{op}(ap'))$ then $(s_{I^*}(ap'') \leqslant s_{I^*}(ap')) \wedge (c_{op}(ap'') > c_{op}(ap'))$

In both cases, we have $ap'' \prec ap'$. □

Thus, if $PAR_{DRF}(B, ap, \mathcal{S}) \neq \emptyset$, adding any action in B to ap cannot yield a Pareto non-dominated solution.

Pareto reduction in *FRF* requires a more complex partial order to define $PAR_{FRF}(B, ap, \mathcal{S})$ but the demonstration is similar. These heuristics are denoted $PAR(B, ap, \mathcal{S})$ when the framework is not specified.

5.3.6. REDUCE function

Finally, we have $\text{REDUCE}(B, ap) =$

$$\begin{cases} B \setminus (LA(B, ap) \cup LC(B, ap) \cup INC(B, ap)) & \text{if } (|ap| \leqslant |N| \wedge PAR(B, ap, \mathcal{S}) = \emptyset) \\ \emptyset & \text{else} \end{cases}$$

5.3.7. Properties of the algorithm

Theorem 19. *Whatever the choice of the reasoning framework among DRF or FRF, and of synthesis function among $\tilde{s}$ and $\hat{s}$, Algorithm 3 terminates and returns a vector $\mathcal{S}$ containing the Pareto non-dominated action plans in the sense of $\prec$, and satisfying conditions $s_{I^*}(ap) \geqslant \alpha_0$ and $c_{\text{op}}(ap) \leqslant budget$.*

Proof. First, condition $(s_{I^*}(ap) \geqslant \alpha_0) \wedge (c_{\text{op}}(ap) \leqslant budget)$ is necessarily satisfied for every element in $\mathcal{S}$ as it corresponds to IS_ADMISSIBLE.

The selected order of actions (function CHOOSE) exerts no influence on any of the algorithm's termination, completeness and consistency properties.

Termination is obvious as we use a branch and bound algorithm and the size of the search space is finite. Thus we just have to show that the algorithm does not deter from reaching a better action plan. To demonstrate the non-deterioration, we just need to prove two proposals:

Assertion 20

Any action plan ap minimal in the sense of the Pareto order on the action plans is necessarily an element of the set of solutions $\mathcal{S}$.

Proof. Let ap be non-dominated. There are several cases in which the branch leading to ap is cut:

- Assume that ap is never reached because of the IS_ADMISSIBLE condition, which cuts the branch that yields ap. This means that there exists $ap' \subset ap$ such that ap' is admissible (IS_ADMISSIBLE(ap', I^*) is true). We need to consider the two frameworks *DRF* and *FRF*:
  - Under *DRF*: as ap' is admissible, we have $s_{I^*}(ap') \geqslant \alpha_0 > 0$. Hence as $ap' \subset ap, s_{I^*}(ap) \leqslant s_{I^*}(ap')$. This holds for both $\tilde{s}$ and $\hat{s}$. Moreover, $c_{\text{op}}(ap) > c_{\text{op}}(ap')$ (see (11)). Therefore

    $$(s_{I^*}(ap'), -c_{\text{op}}(ap')) \prec (s_{I^*}(ap), -c_{\text{op}}(ap)),$$

    which contradicts the fact that ap is Pareto non-dominated.
  - Under *FRF*, ap cannot be blocked because the IS_ADMISSIBLE condition does not prevent us from exploring the subtree.
- Assume that we obtain solution ap which fulfills the IS_ADMISSIBLE condition, but in the instructions after ①, ap is not kept. This implies that there exists $ap' \in \mathcal{S}$ such that $ap \prec ap'$, which contradicts the fact that ap is non-dominated.
- Assume that ap is stored in $\mathcal{S}$, but it is removed from $\mathcal{S}$ later (see instruction after ①). This means that a new admissible action plan ap' dominated ap in the sense of $\prec$, which is again impossible.
- Assume that ap is cut because of the REDUCE function. We are after ②.
  According to Lemmas 15–17, actions in $LA(B, ap), LC(B, ap)$ and $INC(B, ap)$ can be removed from B without changing the Pareto front of solutions.
  Moreover, when $|ap| \geqslant |N|$ or $PAR(B, ap, \mathcal{S}) \neq \emptyset$, adding any action to ap yields action plans that are necessarily dominated in the sense of $\prec$ (see Lemmas 14 and 18) and thus cannot belong to the Pareto front of solutions. Hence reductions in the REDUCE function do not prevent us from obtaining Pareto optimal solutions.

Finally, ap is never cut unless it is stored in $\mathcal{S}$. □

Assertion 21. There is no element in $\mathcal{S}$ that dominates another one in $\mathcal{S}$ in the sense of $\prec$.

Proof. We have to prove that $\mathcal{S}$ only contains elements which are not dominated by another element of $\mathcal{S}$. Algorithm SOLVE is called recursively. We prove this property by induction on the calls of algorithm SOLVE. First of all, the first time the algorithm is called with $\mathcal{S} = \emptyset$, which clearly satisfies the property. Hence we only need to show that if algorithm SOLVE$(B, I^*, ap, \mathcal{S})$ is called, where $\mathcal{S}$ satisfies the property, then $\mathcal{S}'$ that is returned also satisfies the property.

Let us consider a call SOLVE$(B, I^*, ap, \mathcal{S})$ where $\mathcal{S}$ satisfies the property.

First consider the instructions just after ①. Then ap is added to $\mathcal{S}$ if there is no $ap' \in \mathcal{S}$ such that $ap \prec ap'$. In this case, all elements in $\mathcal{S}$ that are dominated by ap are removed. Then $\mathcal{S}'$ is set at either $\mathcal{S}$ or $(\mathcal{S} \cup \{ap\}) \setminus \{ap' \in \mathcal{S} : ap' \prec ap\}$. Thus $\mathcal{S}'$ also satisfies the property.

$\mathcal{S}'$ can also be changed in the instructions after ④. By the induction assumption, as $\mathcal{S}'$ satisfies the property, then $\mathcal{S}''$ also satisfies the property. We repeat this reasoning for the instruction after ⑤. Hence, $\mathcal{S}'$ ultimately satisfies the property. □

When combining Assertions 20 and 21, the result of the algorithm contains exactly all the non-dominated action plans in the sense of $\prec$. □ The following result provides some conditions under which it is certain that Algorithm 1 produces successive improvements (the overall satisfaction cannot decrease).

Corollary 22. *Under DRF and $\hat{s}$, if, after each iteration of "GeneralProcess" in Algorithm 1, the set $\mathcal{S}$ is not empty, then the overall satisfaction $U(T(\gamma))$ can decrease over the iterations.*

Note that this result is also true for Algorithm 2.

Table 1
Domain of digraphs covered and resolution parameters.

Digraph domain				Resolution parameters	
$\|A\|$	$\|N\|$	$\%\|NNA\|$	$\%\|NNAP\|$	α_0	*budget*
10–200 (step by 5)	5–30 (step by 5)	10–50% (step by 5)	25–60% (random)	0.2–0.9	$[1\text{–}5] * \|N\|$

For each $|A|, |N|$, $\%|NNA|$, about 100 digraphs are randomly generated. Then, $\%|NNAP|$ is randomly associated with each digraph.

Proof. Whatever the action plan $ap \in \mathcal{S}$ chosen by the user, all criteria in I^* are satisfied at least to a degree α_0. Under *DRF* and $\hat{s}$, we are sure that no criterion in $N \setminus I^*$ is deteriorated: $\forall i \in N \setminus I^*,\ u_i(T_i(\gamma)) \geqslant u_i(T_i(\gamma^0))$ and $\forall i \in I^*,\ u_i(T_i(\gamma)) > u_i(T_i(\gamma^0))$ where γ^0 is the initial system and γ the system associated with the application of *ap*. Hence, the satisfaction over each criterion is non-decreasing. By the monotonicity condition on the aggregation function *F*, the overall satisfaction $U(T(\gamma)) = F(u_1(T_1(\gamma)), \ldots, u_k(T_k(\gamma)), \ldots, u_n(T_n(\gamma)))$ is non-decreasing from one iteration to the next one: $U(T(\gamma)) > U(T(\gamma^0))$. □

6. Experiments related to the FindActions function efficiency

6.1. Experimental setting and results

Experiments were carried out upon a set of about 156,000 automatically generated problems. Each problem is defined by a digraph structure. $|A|$ ranges from 10 to 200, and $|N|$ from 5 to 30 in the benchmark domain. Let $NNA = \{(a,i) \in A \times N : max(S_i(a), D_i(a)) > 0\}$ (arcs with non nul effect) and $NNAP = \{(a,i) \in A \times N : S_i(a) > 0\}$ (arcs with positive effect). A program was developed to randomly generate digraphs whose parameters are: $|A|, |N|$, the percentage of *NNA*, the percentage of *NNAP* among *NNA*. Finally, several values of α_0 and allocated *budget* were also included in the tests. The range values for all of these parameters are provided in Table 1. The cost of an action *a* was defined as the number of positive arcs that connect it to performances. A minimal influence degree threshold was set at 0.2.

Each generated digraph was then solved for both the *DRF* and *FRF*. Note that the aim of these experiments was to test the algorithm's efficiency instead of the implementation optimization which is beyond the scope of this paper (see perspectives of the section). Consequently, the evaluation of the complexity of the resolution was done by counting the number of times the CHOOSE method is called (number of explored nodes) instead of the execution time which depends on the implementation optimization. This provides an estimation of the speed of the algorithm which is independent of the CPU speed. Let *ntc* denote this number, which is to be compared to a maximum of $2^{|A|+1}$ which is the complexity of the problem to be solved (see Section 5.1). For a given value *limit*, let $uns(limit)$ denote the number of unsolved problems[1] after at most *limit* calls of the CHOOSE method, and $ns(limit)$ the number of problems among $uns(limit)$ where no solution (*i.e.* α_0- admissible action plan with cost lower than *budget*) has been found after *limit* calls of the CHOOSE method.

Remark 23. On a single CPU, $ntc = 10^6$ explorations are performed in less than 10 min for the domain of problems which is covered by the test.

First we observe that the algorithm, when searching for the Pareto front of solutions, gives good results especially for *DRF*. Indeed, 85% of the digraphs are solved with $ntc < 10^3$, and 97.4% are solved with $ntc < 10^6$ (*i.e.* only 4055 problems among 156,000 remain unsolved: $uns(10^6) = 4055$). Finally, only 3291(2.1%) exceed the limit $2 * 10^6 : uns(2 * 10^6) = 3291$ (see Table 2). The results are less relevant with *FRF* but the algorithm is still efficient: 66% are solved with $ntc < 10^3$ and only 2957 among 70,651 (4.2%) problems exceed the limit $2 * 10^6 : uns(2 * 10^6) = 2957$. Relaxation of constraints due to the *FRF* specificities mainly explain these results. This relaxation makes preliminary reductions inefficient and other reductions arise late in the search tree (see Table 3 for more details concerning the preliminary reductions).

If we only focus on finding one solution[2] instead of searching for the Pareto front, the results are much better for both frameworks (See Table 2). Note that for *DRF*, only 22 among 156,000 (0.015%) problems exceed the limit $2 * 10^6$: $ns(2 * 10^6) = 22$ and 268 among 70651 (0.3%) for *FRF*. This also means that the number of problems without solution that remain unsolved is very low since these values (22 resp. 268) are the upper limits respectively for *DRF* (resp. *FRF*). Thus most of the remaining unsolved problems have solutions (more than 99.3% resp. 90%). In the following, we will focus on these.

In Figs. 3–6, several functions are plotted: **choose** is *mean*(*ntc*), **sol**, **min** and **max** are respectively *mean*, *min* and *max* (number of explored solutions), **comb** is $mean\left(\sum_{i=0}^{min(|N|,|B|)} \binom{|B|}{i}\right)$ (where $\sum_{i=0}^{min(|N|,|B|)} \binom{|B|}{i}$ is the upper bound of *ntc* when considering that the cardinality of an action plan shall not exceed $|N|$ according to the Cardinality reduction – see Section 5.3.1), **red** is *mean*(number of reductions) and **canonic** is $mean(2^{|B|+1})$. These figures use logarithmic scale (base 2). Figs. 3 and 4 show that **choose** and **sol** have approximately the same behavior. By examining them in further detail, we find a

[1] Problems for which the algorithm neither proves their inconsistency nor finds the whole Pareto front of solutions after *ntc* calls to CHOOSE.
[2] The algorithm stops when the first admissible solution is found.

Table 2
Pareto resolution and finding one solution speed

	Nb Exp	*limit*	0	10^2	10^3	10^4	10^5	10^6	$2 * 10^6$
DRF	156,000	*uns*	49,848	31,682	23,439	14,818	7079	4055	3291
		ns	49,848	3056	1590	930	607	435	22
FRF	70,651	*uns*	39,713	31,593	23,889	15,336	8340	3781	2957
		ns	39,713	12,522	10,129	6724	4011	2222	268

Table 3
Slope of linear relations.

α_0	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
DRF	0.28	0.63	0.77	0.85	0.92	0.93	0.96	0.98
FRF	0.29	0.31	0.35	0.39	0.48	0.52	0.63	0.80

Values obtained for $\alpha_0 = 0.2$ apply for any $\alpha_0 < 0.2$ as it is the minimal influence degree of each positive arc of the experiments.

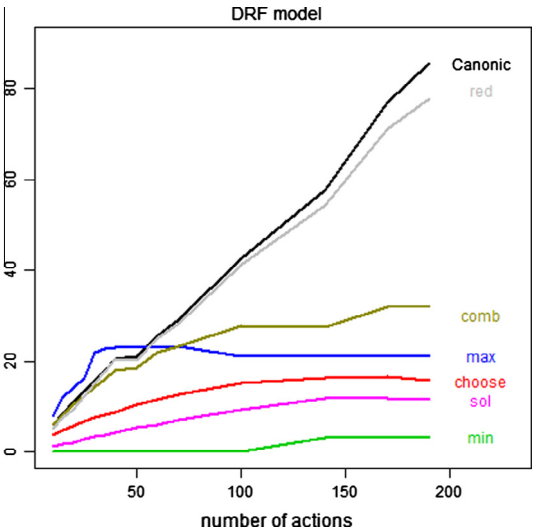

Fig. 3. Behaviors of **choose**, **sol**, **min**, **max**, **comb**, **red** and **canonic** related to the number of actions for *DRF*.

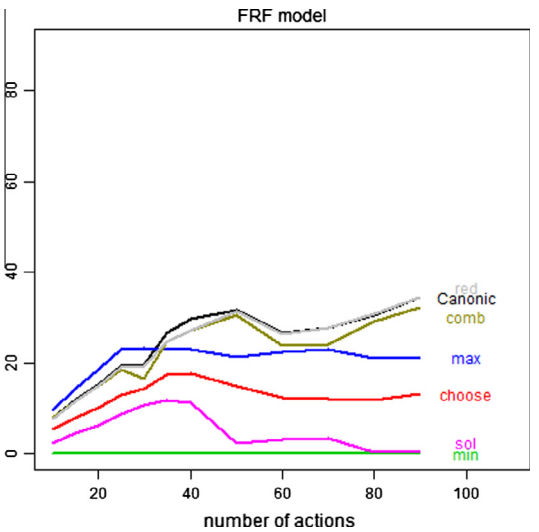

Fig. 4. Behaviors of **choose**, **sol**, **min**, **max**, **comb**, **red** and **canonic** related to the number of actions for *FRF*.

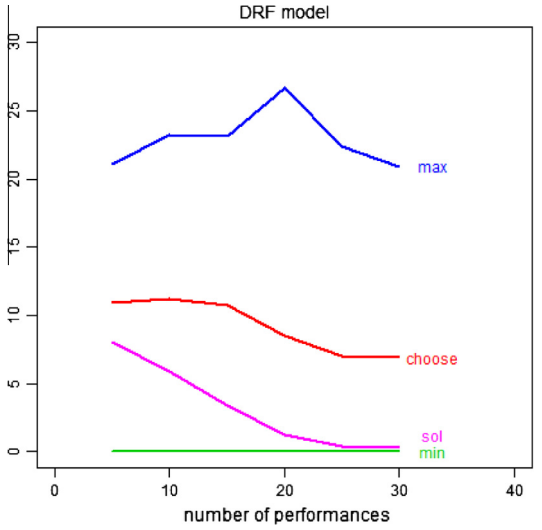

Fig. 5. Behaviors of **choose**, **sol**, **min**, **max**, **comb**, **red** and **canonic** related to the number of performances for *DRF*.

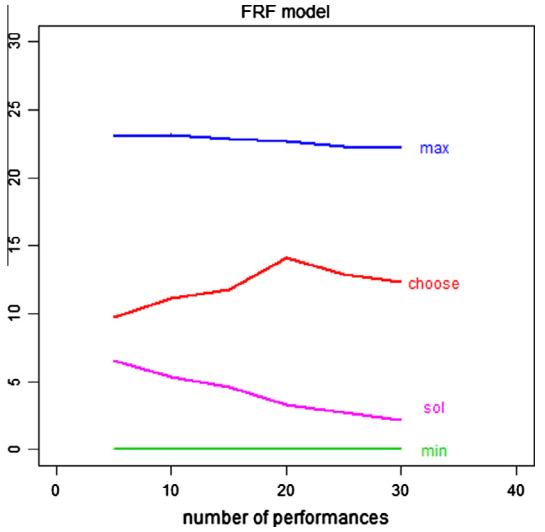

Fig. 6. Behaviors of **choose**, **sol**, **min**, **max**, **comb**, **red** and **canonic** related to the number of performances for *FRF*.

polynomial relation between them such that **choose** $\leqslant$ (**sol**)^k (See Figs. 7 and 8). The average value for k is 1.8 (resp. 1.9) for *DRF* (resp. *FRF*).

Remark 24. We limit the number of performances $|N|$ to a maximum of 30 because, as shown in Figs. 5 and 6, the average complexity (**choose**) of problems decreases when the number of performances increases.

Now, let us examine the efficiency of the algorithm and its heuristics in greater detail. We can distinguish three kinds of reductions:

- *canonical or preliminary reductions*: these preliminary reductions (*Locking Actions* and *Restricting Actions* reductions) are performed before any exploration of the search space and allows us to reduce the problem into its canonical form, which depends on α_0. Several α_0 were tested for both frameworks and in each case the mean number of eliminated actions had a linear relation with the initial number of actions. Table 3 shows the slope of the linear relation for each α_0 (mean percentage of eliminated actions). These reductions may be very significant: *e.g.* in *DRF*, a mean of 63% of the actions are eliminated for $\alpha_0 = 0.3$. This suppresses $2^{|A|} - 2^{0.37*|A|}$ nodes (about $1.6 * 10^{60}$ for $|A| = 200$ with a canonical problem size reduced to about $1.9 * 10^{22}$). For this reason, the efficiency of the heuristics applied after this step are related to the size of the canonical problem (see Figs. 3–6),

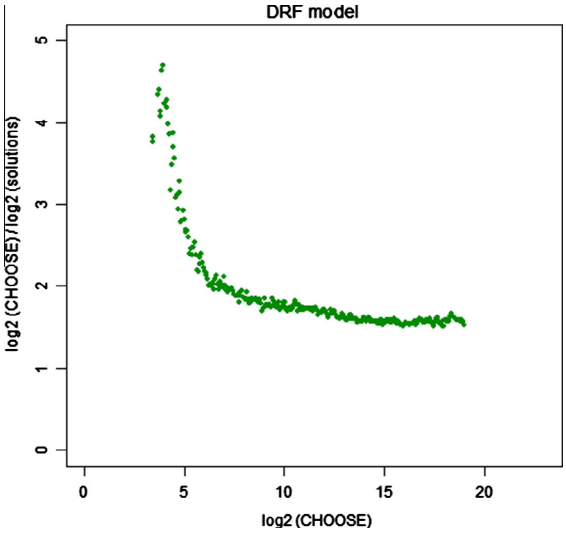

Fig. 7. Relationship between the number of choose and the number of found solutions for *DRF*.

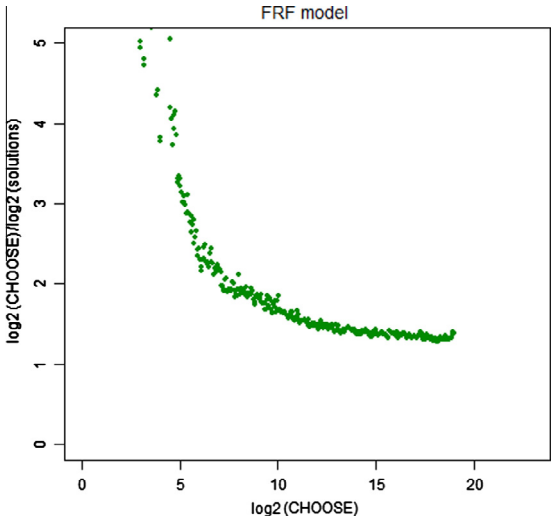

Fig. 8. Relationship between the number of choose and the number of found solutions for *FRF*.

- *structural reductions*: the only one is *Cardinality Reductions*. It is only linked to the size of the problem ($|A|$ and $|N|$). It will reduce the amount of the search space to be explored to $\sum_{i=0}^{min(|N|,|A|)} \binom{|A|}{i}$ instead of $2^{|A|}$. *Cardinality Reductions* are applied during the exploration of the search space and limits the exploration depth. In fact, they are usually not needed during the search because other reductions have already cut the branches before reaching *N*. Figs. 9 and 10 show its impact on the reduction of the search space: it provides the mean number of reductions as a function of $|A|$,
- *semantic reductions*: they consist of *Cost*, *Locking Actions*, *Incompatible Actions* and *Pareto* reductions. Figs. 9 and 10 show their impact on reducing the search space.

At first sight, *Incompatible Actions* and *Locking Actions* seem to be the main contributors to the reductions. In fact, they act as the garbage collector of the algorithm: when complex deductions have suppressed or chosen an action, then they reduce the problem. Thus reductions are mostly attributed to them whereas they are not the genuine initiators. In addition *Cost*, *Cardinality* and *Pareto* reductions occur later during the exploration which limits their contributions.

6.2. Experiment summary

About 100 days of equivalent single CPU computing time were required for the experiments described in this section (30 real days for 3 CPUs). The experiments covered the entire (resp. half) digraph domain of the experiments for *DRF* (resp. *FRF*)

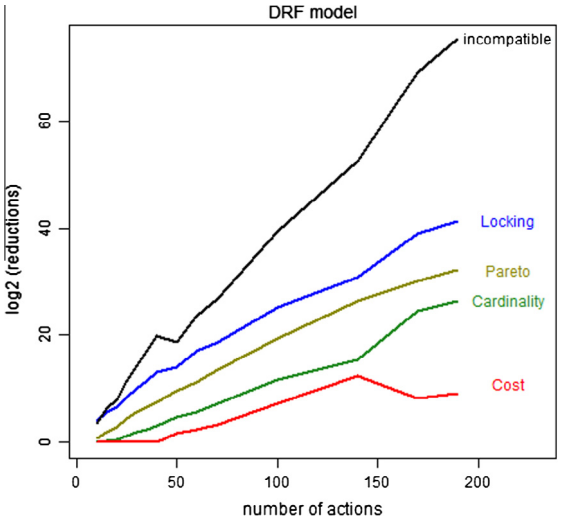

Fig. 9. Mean number of reductions *w.r.t.* $|A|$ for *DRF*.

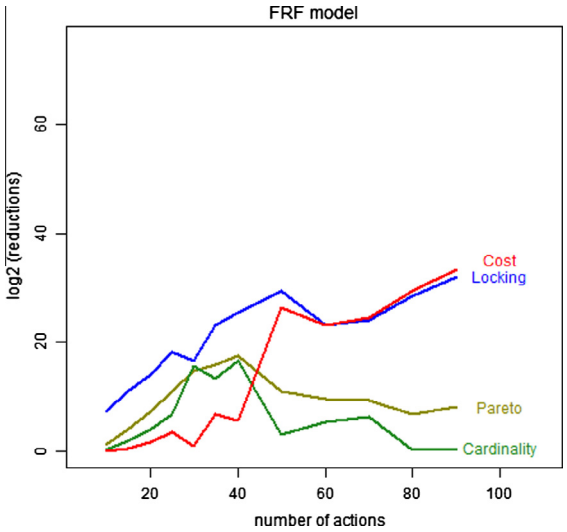

Fig. 10. Mean number of reductions *w.r.t.* $|A|$ for *FRF*.

in accordance with the parameter variations in Table 1. 156,000 digraphs for *DRF* and 70,651 for *FRF*, *i.e.* more than 225,000 were solved. These experiments highlighted the efficiency of the algorithm. Indeed, it did not manage to find at least one solution after $2 * 10^6$ iterations (*ntc*) for only $\frac{(22+268)}{226,651} = 1.28 * 10^{-3}$. Furthermore, for most of these problems, the whole set of solutions (the Pareto front) was found in less than $2 * 10^6$ iterations. This is to be compared to the real complexity of the problem which initially ranges from $2^{50} \approx 10^{15}$ for the smallest problems to $2^{200} \approx 10^{60}$ for the biggest ones.

Several directions may be considered to further enhance the algorithm's performances, as detailed in the final conclusion.

7. Case study

Our case study concerns the adaptive management of a manufacturing plant. Strategic improvement priorities evolve over time, depending on the competitive context, and new improvement actions must be continuously planned to compensate for contextual disturbances. This illustrative example has not yet been conducted in a real enterprise setting. It may thus have a lack of some industrial constraints. For example, certain constraint relationships between actions are concealed in the model because they would require further knowledge regarding industrial improvement methods.

The improvement actions specified in this case study may be improvement methods (*e.g. Kanban*) or strategic schedules (*e.g. staff training*). This ambiguity could be criticized from a purely industrial engineering standpoint, although a full understanding of this illustrative example does not require any further attention.

Table 4
Influence matrix.

Improvement method	*SC*	*TD*	*PP*	*RP*	*PQ*
Kanban	−0.5	+0.5	+0.9	+0.6	+0.2
SMED	−0.5	+0.8	0.	+0.7	+0.3
POK	+0.5	+0.2	−0.3	−0.4	+0.6
6Σ	−0.2	+0.5	−0.2	0.	+0.8
TQM	+0.1	−0.3	−0.2	0.	+0.9
TPM	0.	+0.8	+0.3	+0.4	0.
SRM	+0.6	+0.7	−0.4	+0.7	0.
PR	−0.8	0.	+0.9	0.	−0.5
ST	+0.9	0.	−0.6	+0.5	+0.7
TP	+0.3	+0.2	−0.2	−0.3	+0.8
CNP	0.	−0.3	−0.5	−0.4	+0.8

The company's overall objective is to increase customer satisfaction. On the one hand, four criteria are established by the company to assess its overall satisfaction: *RP*: *Product Range*, *PP*: *Product Pricing*, *PQ*: *Product Quality*, and *TD*: *Delivery Time*; these are then complemented by an internal criterion: *SC*: *Social Climate*. On the other hand, actions involved in the relationships with such criteria are possible, which may correspond to implementing industrial performance improvement methods. First, six classical improvement methods will be examined: *Kanban* (*Kanban* is related to lean and just-in-time production), *SMED* (Single-Minute Exchange of Die provides a rapid and efficient approach to convert a manufacturing process from producing the current product to producing the next product), *Poka-Yoke* [35] (denoted POK – a Poka-Yoke is any mechanism within a lean manufacturing process that helps an equipment operator avoid mistakes; it aims to eliminate product defects by preventing, correcting or signaling human errors as they occur), *Six Sigma* (denoted 6Σ – Six Sigma seeks to improve the quality of process outputs by identifying and removing the causes of defects and minimizing variability in manufacturing and business processes), *TQM* (Total Quality Management is a set of management practices introduced throughout the organization and geared to ensuring that the organization consistently meets or exceeds the customer requirements), and *TPM* (Total Predictive Maintenance is a new way of looking at maintenance, according to a proactive approach that is essentially aimed at preventing any slack before it occurs). Strategic schedules can then be envisaged: *SRM* (supplier relationship management), *PR* (partial relocation in search of less expensive labor), *ST* (staff training), *TP* (traceability policy), and *CNP* (respect of standards and conformity with new policies).

These six industrial improvement methods and five strategic schedules define the set of actions ($|A| = 11$) in our framework. They may be combined with design action plans, as proposed in Section 4 to improve the satisfaction of the industrial system. In this case study, it is assumed that these actions are not mutually exclusive and that there is no constraint regarding their joint application, their respective response time and precedence temporal rules: they may be carried out independently of one another to improve the criteria. This assumption might be discussed from an operational standpoint but would induce further difficulties in our approach. Indeed, additional constraints would merely reduce the combinatorial of action plan computation on one hand, whereas additional criteria could be introduced s.t. "time to produce the effect" to take temporal requirements into account on the other hand. In our framework, it is assumed that each improvement method or strategic schedule may be associated with an industrial system parameter but it is not necessary to specify this connection.

Table 4 provides the influence Influence(a, i) of actions over criteria (see (6)). As an example, action *PR* should probably have a positive impact on *PP*, whereas it should clearly have a negative impact on *SC* within the company.

Let us suppose that the company's overall satisfaction is defined as the aggregation of the individual satisfaction degrees with respect to the five criteria (see Section 2.1), where F is a Choquet integral aggregation function [23]. The family of Choquet integrals provides aggregation functions that accommodate both the relative weighting of criteria and their interactions; it covers a wide range of preference models. In this application, we consider a particular case of Choquet integrals, based on the so-called 2-additive measure [26,21]: according to this simplified model, only interactions by pairs of criteria are considered. The 2-additive Choquet integral can then be expressed for an element $(y_1, \ldots, y_n) \in \mathbf{R}^n$ interpretable form as follows [36]:

$$CI(y_1, \ldots, y_n) = \sum_{i=1}^{n} v_i \cdot y_i - \frac{1}{2} \sum_{i>j} I_{ij} \cdot |y_i - y_j|$$

with the property that $((v_i - \frac{1}{2}\sum_{i \neq j} |I_{ij}|) \geqslant 0)$ and where the v_i are the Shapley indices, representing the importance of each criterion relative to all the others, where $\sum_{j=1}^{n} v_1 = 1; I_{ij}$ denotes the interactions between pairs of criteria (i, j) with values contained in the interval $[-1, 1]$. A value 1 indicates full complementarity between the two criteria (which are expected to be simultaneously satisfied), a value of -1 indicates full redundancy, and a null value means that the criteria are independent.

Table 5 lists the company's initial vector of satisfaction degrees $u(T(\gamma^0))$, along with the Shapley index of each criterion. The approximate average cost required to improve the satisfaction degree from 0 to 1 (uc^i_{str}) for each criterion is also provided in Table 5. The interactions between criteria are given in Table 6. In the context relative to $u(T(\gamma^0))$, the preference

Table 5
Weights, costs and initial satisfaction degrees.

	Indicators	Shapley index(v_i)	uc^i_{str}	$u(T(\gamma^0))$
1	*Social Climate* (*SC*)	0.10	1000k€	0.7
2	*Delivery Time* (*TD*)	0.10	3000k€	0.8
3	*Product Pricing* (*PP*)	0.25	4000k€	0.4
4	*Product Range* (*RP*)	0.05	2000k€	0.7
5	*Product Quality* (*PQ*)	0.50	4000k€	0.4

Table 6
Interaction coefficients.

Improvement method	*SC*	*TD*	*PP*	*RP*	*PQ*
SC	0.	0.	0.	0.	0.
TD	0.	0.	0.	0.05	0.1
PP	0.	0.	0.	0.05	0.45
RP	0.	0.05	0.05	0.	0.
PQ	0.	0.1	0.45	0.	0.

Table 7
$\omega_I(CI)(u(T(\gamma^0)))$ for several coalitions I.

I	$\omega_I(CI)(u(T(\gamma^0)))$
$\{PP, PQ\}$	1.0137E−4
$\{SC, PP, PQ\}$	1.0129E−4
$\{SC\}$	1E−4

Table 8
Pareto front for all possible choices among *DRF*/*FRF* and $\tilde{s}_{I^*}(ap)/\hat{s}_{I^*}(ap)$. For each Pareto set, we show in brackets the vector $(s_{I^*}(ap), -c_{\text{op}}(ap))$.

	DRF	*FRF*
$\tilde{s}_{I^*}(ap)$	{*Kanban*} $(0.2, -1)$ {*TQM,PR*} $(0.9, -2)$	{*Kanban*} $(0.2, -1)$ {*TQM,PR*} $(0.9, -2)$ {*Kanban,TQM*} $(0.9, -2)$
$\hat{s}_{I^*}(ap)$	{*PR,ST*} $(0.7, -2)$	{*PR,ST*} $(0.7, -2)$ {*Kanban,ST*} $(0.7, -2)$ {*Kanban,TQM,SRM*} $(0.9, -3)$ {*Kanban,TQM,ST*} $(0.9, -3)$

model embedded in the Choquet integral could be synthesized with a rule of the following type: the company pays special attention to quality and product pricing policies, yet these policies must be accompanied by adequate services relative to the product range and delivery time in order to be truly attractive (positive interactions). Moreover, the company must preserve its social climate (independence). In this illustrative example, we assume that all actions yield the same cost, say $c_{\text{op}}(a) = 1$ for all $a \in A$. We assume that the decision maker's expectation is $U_{\min} = 0.85$. Moreover, $\alpha_0 = 0.2$ and $budget = 5$.

Let us look at the first two iterations of Algorithm 1.

Iteration 1 starting from γ^0:

- $u(T(\gamma^0)) = (0.7, 0.8, 0.4, 0.7, 0.4)$ and thus $U(T(\gamma^0)) = CI(u(T(\gamma^0))) = 0.454$. As $U(T(\gamma^0)) < U_{\min}$, we need to find a suitable action plan to improve the satisfaction degrees $u_i(T_i(\gamma^0))$.
- FindCoalition$(2^N, u(T(\gamma^0)))$: For profile $u(T(\gamma^0))$, the coalition on which $\omega_I(CI)(u(T(\gamma^0)))$ is the largest is $\{PP, PQ\}$ (cf. Table 7). This result is completely natural as these two criteria have the worst scores and also the greatest importance. Moreover, there is a strong positive interaction among them. Hence, it is more rewarding to improve both *PP* and *PQ* rather than any of them individually ($\omega_{\{PP\}}(CI)(u(T(\gamma^0))) = 1.058\text{E}{-5}$ and $\omega_{\{PQ\}}(CI)(u(T(\gamma^0))) = 7.967\text{E}{-5}$). The recommendation to improve $\{PP, PQ\}$ is just ahead of that for *SC* alone because of the relative costs of improving *PP* and *PQ*.
- FindActions(A, I^*) with $I^* = \{PP, PQ\}$. The results of the algorithm SOLVE (Algorithm 3) are presented in Table 8 for all possible choices among *DRF* vs. *FRF* and $\tilde{s}$ vs. $\hat{s}$. Assume that the decision maker chooses to implement *TQM,PR*, which is supposed to improve the impact on *PP*, *PQ*, but slightly decrease the satisfaction on *SC*, *TD*.

Table 9
$\omega_I(CI)(u(T(\gamma^1)))$ for several coalitions I.

I	$\omega_I(CI)(u(T(\gamma^1)))$
$\{SC\}$	1E−4
$\{SC, PP, PQ\}$	8.75E−5
$\{PP, PQ\}$	8.4375E−5

Iteration 2 starting from γ^1:

- $u(T(\gamma^1)) = (0.6, 0.7, 0.8, 0.7, 0.8)$ and thus $U(T(\gamma^1)) = CI(u(T(\gamma^1))) = 0.7575$. As $U(T(\gamma^1)) < U_{\min}$, we need to find a suitable action plan to improve the satisfaction degrees $u_i(T_i(\gamma^1))$.
- FindCoalition$(2^N, u(T(\gamma^1)))$: $\omega_I(CI)(u(T(\gamma^1)))$ is maximal for SC (see Table 9).
- FindActions(A, I^*) with $I^* = \{SC\}$. This time, the decision maker decides to use *DRF* and $\hat{s}$ (he does not want any decrease in satisfaction on the criteria). The application of algorithm SOLVE gives *Kanban,ST* (with $(0.9, -2)$) and *PR,ST* (with $(0.9, -2)$).

Once the improvement becomes more complex, *i.e.* greater number of criteria to be improved simultaneously, it becomes obvious that the *Optimistic Attitude* offers a broader range of admissible solutions. Furthermore, these solutions are considered with higher degrees of admissibility in the latter case.

8. Related works

As described in Section 2, two challenges arise when designing a system improvement: at the strategic level, where changes in system outputs would be expected to bring significant improvements, and at the operational level, where system parameter adjustments should be carried out in order to achieve the expected improvement.

In the Industrial Engineering literature, industrial performance management generates a considerable body of work relying on preference models. A Performance Measurement System (PMS) consists of a multi-criteria instrument for informing and supporting decision-makers on diagnosis and improvement activities [4,7,42,32]. From a global standpoint, a PMS is a multi-criteria instrument for informing decision-makers about a variety of different things, *e.g.* the performance level, the reasons for poor or good performance, and the criteria for which improvement is required. A PMS consists of a set of performance expressions to be consistently organized with respect to the company's objectives. For instance, the objectives/performances of the manufacturing workshops contribute to the objectives/performances of the manufacturing plants, which in turn contribute to the objectives/performances of the company [4].

Given its nature, a PMS thus requires multi-criteria methods [50]. It solely focuses on strategic management considerations and goal-based reasoning. Hence, an alternative standpoint can be found in the Industrial Engineering literature. Felix criticized the way preference models were established for the purpose of designing improvement projects [11]. According to his mindset, a preference model must capture the cooperative or competitive nature of goals.

Another thorny issue encountered when designing a complex system improvement pertains to the transformation that provides expected improvements relative to the system goals obtained from an input parameter vector.

The qualitative model describing relationships between input parameters and goals can be represented by a causal digraph [39]. Many causality definitions are provided in the literature which generally consider two concepts of causality that can occur in physical systems: dynamical causality and instantaneous coupling [2]. A vast body of literature exists on the use of causal digraphs to describe oriented relationships between inputs (*e.g.* input parameters) and goals within complex systems [14]. For example, [14] proposes a Bayesian network to define the risk factors and their causal relationships, and then perform security vulnerability propagation analysis in information system security, to determine the propagation paths with the highest probability and the greatest estimated risk value. These relationships constitute relevant models for both cognitive and explanatory purposes. Cognitive Maps (CM) offer one such trend [45]. Because of their user-friendly semantics, qualitative causal digraphs appear to be appropriate models for capturing the knowledge available pertaining to the complex input parameters-to-goals transformation in industrial settings.

The general action concept may be preferred to the parameter adjustment concept in the literature. The influence associated with an arc thus corresponds to the notion that actions either support or hinder the achievement of a goal. The purely qualitative representation of action–goal relationships is proposed in [39]. For cases when the influence of actions can be more accurately characterized, a fuzzy relationship model has been proposed by Felix [11,12].

The next issue to be addressed pertains to how the influences of actions on a specific goal are combined in an imprecise and/or uncertain context. Published studies on this issue can be found in Requirements Engineering [35] and Software Engineering [20,19]. In these goal-driven models, the relation derived between an action a and a goal i may be expressed by degrees of satisfaction and denial for a given goal may be chosen in a qualitative setting [9,19,20]. When several actions for goal i exist in the digraph, the overall degree of satisfaction (resp. denial) for i is equivalent to the conjunction of degrees of satisfaction (resp. the disjunction of degrees of denial proof) of the goal's previous actions in the digraph. This notion is

close to those presented in [40,39], whose authors combined this goal-driven representation with Felix's goal relationship model [11,12].

Other works in the field of Model-driven Engineering also require a model of the qualitative impacts between actions on parameters and system goals, *e.g.* the Dynamic Adaptive Systems (DAS) community [18]. Since many systems need to be adapted to an evolving context, Model-driven Engineering approaches are aimed at describing models relative to both their design time and runtime, thus providing possible DAS configurations. In [17,18], a Domain-Specific Modeling Language (DSML) has been proposed to represent variability in a system. This language is more general than our representation of all potential actions. The main objective in all of these works can be summarized as the search for an adequate parameter configuration that improves the level of satisfaction of a set of goals, possibly prioritized or interacting, whenever the known relationships between actions and goals are imprecise impacts or uncertain influences.

Our approach differs from others existing approaches in the following way. First, it may be confusing to associate preferential interactions between goals and behavioral influences between actions and goals from a semantic standpoint. This approach may result in misleading interpretations as regards the improvement project. Secondly, it may be simpler in practice to break down the improvement process into steps and then distinguish preferential goal-based interactions and action–goal relationships when the system is overly complex. The formal model presented in Section 2 illustrates this "will vs. act" breakdown and supports the viewpoint defended in this paper.

9. Conclusion

This paper proposes a trade-off between managerial and implementation aspects of industrial improvements. The formal model detailed in Section 2 distinguishes which choices during the design of an improvement project depend on the operational constraints of the required system from choices related to the designer's/operator's strategic preferences. Based on this framework, the other sections were dedicated to mathematically supporting the search for an efficient improvement, as required in a complex system.

The MAUT model enables us to synthesize managerial preferences with respect to the criteria to be improved first. Managerial preferences have been recorded in an analytical form that facilitates the search for strategic improvements relative to optimization problems, which therefore provides a powerful artifact for recording overall company satisfaction and deriving a rationale from a managerial perspective. The MAUT model thus helps identify the way in which a system should appropriately evolve in order to improve its satisfactions to the greatest extent possible. The preference model is then complemented by other models that take the operational context into account. More specifically, a model of relationships between the system's assessment criteria and its input parameters is needed to successfully complete the improvement implementation component. Our paper thus proposed a fuzzy model in Section 4. The fuzzy digraph not only captures the system's behavioral constraints, but it also includes the step of interpreting system attributes in terms of satisfaction degrees. It must not be considered therefore as a classical behavioral model of a system since its outputs depend on the designer's/operator's risk adversity and his optimistic vs. pessimistic behavior regarding the impacts of actions he intends to carry out in order to achieve the expected satisfaction. Subjectivity was thus introduced in the behavioral model.

It has also been explained herein how both models must be used in an iterative procedure to design an efficient system improvement: qualitative knowledge about the system prevents assessing the precise impacts of improvement actions in a deterministic manner. The iterative improvement process is thus supported by a mathematical model, in addition to a software tool that allows testing of our approach on an industrial case study.

In conclusion, the way in which these models are jointly used within our entire design procedure was intended to show that both models should be used in tandem in order to address managerial and implementation issues involved in an improvement project. From a modeling standpoint, this framework allows standardization of the formalisms of a large body of work on complex system control or improvements stemming from many communities, such as goal-driven engineering, industrial engineering and adaptive systems. From a pragmatic perspective, the challenge herein consists of developing a consensual transition from motivation to action, between managerial decisions and operational capabilities. As mentioned previously, the notion beyond this framework is that in any improvement project, executive managers are seeking the best for their companies, while operatives are doing their best. Meeting this challenge was the basis of our proposal.

Concerning the experiments, several directions may be considered to further enhance the algorithm's performances. First, we aim at introducing new heuristics to improve the results and reduce the number of unsolved problems, especially for *FRF* (*e.g.* Pareto solutions obtained from *DRF* could be used to initialise the *FRF* resolution). Then we intend to work on optimization of the implementation. The ongoing complexity of the algorithm is linear in space[3] $\mathcal{O}(|A| * |N|)$ but quadratic in time for each call to CHOOSE or REDUCE $\mathcal{O}(|A|^2 * |N|)$. This means that its time complexity is about $\mathcal{O}(|A|^2 * |N|) * ntc$. Our studies on the algorithm show that it could be reduced in time to $\mathcal{O}(|A| * |N|) * ntc$ by preprocessing and sorting information of the digraph (Actions, Arcs and Criteria). This one shot preprocessing could be done in $\mathcal{O}(|A|^2 * |N|)$ in time and space. This should significantly enhance the resolution time: instead of about 10 min to perform $ntc = 10^6$, 5 s would be sufficient with the same CPU.[4]

[3] When $|A| > |N|$.

[4] But will make implementation much more complex.

Appendix A. Main notations

Variable	Definition
$\gamma = (\gamma_1, \ldots, \gamma_p) \in \Gamma$	Input parameters configuration of the system
Γ	Set of all possible configurations
$\Gamma_{Adm} \subseteq \Gamma$	Set of all feasible configurations
$X_1, \ldots, X_n$	Set of attributes (observable outputs) of the system
$X = X_1 \times \cdots \times X_n$	Set of alternatives, described by one value for each attribute
$T : \Gamma_{Adm} \rightarrow X$ $T(\gamma) = (T_1(\gamma), \ldots, T_n(\gamma))$	Transformation that yields the values on system's attributes obtained from a configuration
$x_i = T_i(\gamma) \in X_i$	Impact of configuration γ on attribute i
$N = \{1, \ldots, n\}$	Set of n criteria
$u_i : X_i \rightarrow [0, 1]$	Utility function related to attribute i
$u_i(x_i)$	The extent to which the goal associated with criterion i is satisfied by x_i
$F : [0, 1]^n \rightarrow [0, 1]$	Aggregation function
$U(x) = U(T(\gamma))$ $U(x) = u(T(\gamma))$	Aggregated utility of configuration γ (overall satisfaction of configuration γ)
$u(T(\gamma)) \in [0..1]^n$	Vector of the satisfaction degrees on the criteria
$I, I^* \subseteq N$	Subsets of criteria on which an alternative needs to be improved
$\omega_I(F)(u(T(\gamma))$	Worth index quantifying the worth for $u(T(\gamma))$ to be improved in criteria among I subject to the evaluation function F
A	Set of valid actions (*e.g.* increasing a parameter's value)
$a \in A$	Elementary improvement action
$AP \subseteq 2^A$; ap	Set of action plans; one action plan (a subset of actions)
$S_i(a), D_i(a),\ a \in A$	Degree of belief to which a can positively (resp. negatively) affect criterion i
DRF	Drastic Reasoning Framework
FRF	Flexible Reasoning Framework
$s_i(ap), s_I(ap)$	Degree of belief to which ap should contribute to the improvement of i (resp. I)
$c_{op}(a), c_{op}(ap)$	Cost of action or action plan
α	Admissibility degree (expected degree of belief)
LA	Locking Actions
INC	Incompatible Actions
ntc	Number of times the CHOOSE method is called

References

[1] J.M. Alonso, L. Magdalena, Special issue on interpretable fuzzy systems, Inform. Sci. 181 (20) (2011) 4331–4339.
[2] P.-O. Amblard, O. Michel, Causal conditioning and instantaneous coupling in causality graphs, Inform. Sci. (January) (2014).
[3] A. Baykasoglu, A. Oztas, E. Ozbay, Prediction and multi-objective optimization of high strength concrete parameters via soft computing approaches, Expert Syst. Appl. 36 (2002) 6145–6155.
[4] L. Berrah, G. Mauris, J. Montmain, Diagnosis and improvement indexes for a multi-criteria industrial performance synthesized by a Choquet integral aggregation, Int. J. Manage. Sci., OMEGA 36 (2008) 340–351.
[5] L. Berrah, J. Montmain, G. Mauris, V. Clivillé, Optimising industrial performance improvement within a quantitative multi-criteria aggregation framework, Int. J. Data Anal. Tech. Strategies 3 (2011) 42–65.
[6] A. Bilbao-Terol, M. Arenas-Parra, V. Cañal Fernández, Selection of socially responsible portfolios using goal programming and fuzzy technology, Inform. Sci. 189 (2012) 110–125.
[7] U. Bititci, Modelling of performance measurement systems in manufacturing enterprises, Int. J. Prod. Econ. 42 (1995) 137–147.
[8] I. Boussaïd, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, Inform. Sci. 237 (2013) 82–117.
[9] L. Chung, B. Nixon, E. Yu, J. Mylopoulos, Non-Functional Requirements in Software Engineering, Kluwer Acad. Publ., 2000.
[10] L.-C. Fang, S.-Y. Qin, Concurrent optimization for parameters of powertrain and control system of hybrid electric vehicle based on multi-objective genetic algorithms, in: International Joint Conference SICE-ICASE, Busan, South Korea, 2006, pp. 2424–2429.
[11] R. Felix, Relationships between goals in multiple attribute decision making, Fuzzy Sets Syst. 67 (1994) 47–52.
[12] R. Felix, Multicriteria decision making (mcdm): management of aggregation complexity through fuzzy interactions between goals or criteria, in: Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU), Malaga, Spain, June 22–27, 2008.
[13] R. Felix, Multi-goal aggregation of reduced preference relations based on fuzzy interactions between decision goals, in: J.P. Carvalho, D. Dubois, U. Kaymak, J.M. da Costa Sousa (Eds.), Proceedings of the Joint 2009 IFSA/EUSFLAT Conference, Lisbon, Portugal, July 20–24, 2009, pp. 1004–1008.
[14] N. Feng, H.J. Wang, M. Li, A security risk analysis model for information systems: causal relationships of risk factors and vulnerability propagation analysis, Inform. Sci. 256 (2014) 57–73.
[15] P. Fishburn, Utility Theory for Decision-Making, John Wiley & Sons, New York, 1970.
[16] P. Fishburn, The Foundations of Expected Utility, Reidel, Dordrecht, 1982.
[17] F. Fleurey, V. Delhen, N. Bencomo, B. Morin, J. Jézéquel, Modeling and validating dynamic adaptation, in: 3rd Int. workshop on Models @Runtime (MRT'08), at MoDELS'08, Toulouse, France, 2008.
[18] F. Fleurey, A. Solberg, A domain specific modeling language supporting specification, simulation and execution of dynamic adaptive systems, in: 12th Int. conf. Model-Driven Engineering Languages and Systems (MoDELS'09), Denver, Colorado, USA, 2009, pp. 606–621.

[19] P. Giorgini, J. Mylopoulos, E. Nicchiarelli, R. Sebastian, Reasoning with goal models, in: 21st International Conference on Conceptual Modeling, October 2002, pp. 167–181.
[20] B. Gonzalez-Baixauli, J. do Prado Leite, J. Mylopoulos, Visual variability analysis for goal models, in: 12th IEEE International Conference on Requirements Engineering (RE), September 6–11, 2004, pp. 198–207.
[21] M. Grabisch, k-order additive discrete fuzzy measures and their representation, Fuzzy Sets Syst. 92 (1997) 167–189.
[22] M. Grabisch, S. Greco, M. Pirlot, Bipolar and bivariate models in multicriteria decision analysis: descriptive and constructive approaches, Int. J. Intell. Syst. 23 (2008) 930–969.
[23] M. Grabisch, C. Labreuche, A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid, Ann. Oper. Res. 175 (1) (2010) 247–290.
[24] M. Grabisch, J.-L. Marichal, R. Mesiar, E. Pap, Aggregation functions: construction methods, conjunctive, disjunctive and mixed classes, Inform. Sci. 181 (1) (2011) 23–43.
[25] M. Grabisch, J.-L. Marichal, R. Mesiar, E. Pap, Aggregation functions: means, Inform. Sci. 181 (1) (2011) 1–22.
[26] M. Grabisch, M. Roubens, The application of fuzzy integrals in multicriteria decision making, Eur. J. Oper. Res. 89 (1996) 445–456.
[27] M. Hatch, R. Badinelli, Concurrent optimization in designing for logistics support, Eur. J. Oper. Res. 115 (1) (1999) 77–97.
[28] M. Hatch, R. Badinelli, A concurrent optimization methodology for concurrent engineering, IEEE Trans. Eng. Manage. 46 (1) (1999) 72–86.
[29] R. Kazman, M. Klein, P. Clements, ATAM: method for architecture evaluation, Technical Report, CMU/SEI-2000-TR-004, 2000. <http://www.sei.cmu.edu/reports/00tr004.pdf>.
[30] R.L. Keeney, H. Raiffa, Decisions with Multiple Objectives Preferences and Value Tradeoffs, Cambridge University Press, 1976.
[31] D. Krantz, R. Luce, P. Suppes, A. Tversky, Additive and polynomial representations, Foundations of Measurement, vol. 1, Academic Press, 1971.
[32] P. Kueng, A. Krahn, Building a process performance measurement system: some early experiences, J. Sci. Ind. Res. 58 (1999) 149–159.
[33] C. Labreuche, Determination of the criteria to be improved first in order to improve as much as possible the overall evaluation, in: Int. Conf. on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU), Perugia, Italy, July 4–9, 2004, pp. 609–616.
[34] T.-F. Liang, Application of fuzzy sets to manufacturing/distribution planning decisions in supply chains, Inform. Sci. 181 (4) (2011) 842–854.
[35] M.E.R.F. Lopes, C.H.Q. Forster, Application of human error theories for the process improvement of requirements engineering, Inform. Sci. 250 (2013) 142–161.
[36] J.-L. Marichal, Aggregation of interacting criteria by means of the discrete Choquet integral, in: T. Calvo, G. Mayor, R. Mesiar (Eds.), Studies In Fuzziness And Soft Computing, vol. 97, Physica Verlag, Heidelberg, 2002.
[37] L. Martínez, F. Herrera, Challenges of computing with words in decision making, Inform. Sci. 258 (2014) 218–219.
[38] K. Misra (Ed.), Handbook of Performability Engineering, Springer, 2008.
[39] J. Montmain, V. Clivillé, L. Berrah, G. Mauris, Preference and causal fuzzy models for manager's decision aiding in industrial performance improvement, in: FUZZ-IEEE 2010, Barcelona, Spain, 2010.
[40] J. Montmain, C. Labreuche, Amélioration multicritère d'options dans les systèmes complexes, in: LFA'2009: Rencontres Francophones sur la Logique Floue et ses Applications, Annecy, France, 2009.
[41] O. Mouelhi, P. Couturier, T. Redarce, A hybrid search algorithm for multicriteria optimization and evaluation in mechatronic products design, in: Advanced Intelligent Mechatronics, Singapore, 2009.
[42] A. Neely, The performance measurement revolution: why now and what next?, Int J. Oper. Prod. Manage. 19 (1999) 205–228.
[43] A. Osseiran, Qualitative Bayesian networks, Inform. Sci. 131 (2001) 87–106.
[44] Z. Pei, Rational decision making models with incomplete weight information for production line assessment, Inform. Sci. 222 (2012) 696–716.
[45] A. Pena, H. Sosso, A. Guttierez, Causal knowledge and reasoning by cognitive maps: pursuing a holistic approach, Expert Syst. Appl. 35 (2008) 2–18.
[46] H.V. Pham, E.W. Cooper, T. Cao, K. Kamei, Hybrid Kansei-SOM model using risk management and company assessment for stock trading, Inform. Sci. 256 (2011) 8–24.
[47] J. Pignon, C. Labreuche, A methodological approach for operational and technical experimentation based evaluation of systems architectures, in: Int. Conference on Software & Systems Engineering and their Applications (ICSSEA), Paris, France, December 4–6, 2007.
[48] G. Reynoso-Meza, X. Blasco, J. Sanchis, J.M. Herrero, Comparison of design concepts in multi-criteria decision-making using level diagrams, Inform. Sci. 221 (2012) 124–141.
[49] B. Roy, Decision aiding today: what should we expect?, in: T. Gal, T. Stewart, T. Hanne (Eds.), Multicriteria Decision Making: Advances in MCDM Models, Algorithms, Theory and Applications, Kluwer Academic Publishers, Boston, 1999, pp. 1.1–1.35.
[50] S. Santos, V. Belton, S. Howick, Adding value to performance measurement by using system dynamics and multi-criteria analysis, Int. J. Oper. Prod. Manage. 22 (2002) 1246–1272.
[51] I. Tseng, K. Kotovsky, J. Cagan, Concurrent optimization of computationally learned stylistic form and functional goals, J. Mech. Des. 134 (11) (2012) 111016-1–111016-11.
[52] H.-B. Yan, V.-N. Huynh, T. Ma, Y. Nakamori, Non-additive multi-attribute fuzzy target-oriented decision analysis, Inform. Sci. 240 (2013) 21–44.
[53] M. Yoshimura, Concurrent optimization of product design and manufacture, in: H. Parsaei, W. Sullivan (Eds.), Concurrent Engineering: Contemporary Issues and Modern Design Tools, Chapman & Hall, 1993, pp. 343–369.
[54] L.A. Zadeh, Toward a generalized theory of uncertainty (gtu)—-an outline, Inform. Sci. 172 (1) (2005) 1–40.
[55] C. Zhang, H. Wang, Concurrent optimization of design and manufacturing tolerances, in: H. Parsaei, W. Sullivan (Eds.), Concurrent Engineering: Contemporary Issues and Modern Design Tools, Chapman & Hall, 1993, pp. 343–369.