J Heuristics (2006) 12: 329–346
DOI 10.1007/s10732-006-8075-2

Integration and propagation of a multi-criteria decision making model in constraint programming

F. Le Huédé · M. Grabisch · C. Labreuche · P. Savéant

Abstract In this paper we propose a general integration scheme for a Multi-Criteria Decision Making model of the Multi-Attribute Utility Theory in Constraint Programming. We introduce the Choquet integral as a general aggregation function for multi-criteria optimization problems and define the Choquet global constraint that propagates this function during the Branch-and-Bound search. Finally the benefits of the propagation of the Choquet constraint are evaluated on the examination timetabling problem.

Keywords Multi-criteria optimization · Constraint programming · Multi-criteria decision making

1. Introduction

The practice and developments of Constraint Programming (CP) and Operations Research have shown that, in many cases, two complex issues have to be taken into account when addressing real-life optimization problems. First, industrial problems are often highly combinatorial and it is often difficult to solve them fast. Secondly, the preference relation between solutions to a problem is generally complex and depends on several conflicting criteria.

As a result, several formalisms and frameworks have been introduced in CP to handle complex preference relations in optimization problems. For example the Soft Constraints, Preference Based Search, CP-nets frameworks propose various approaches to model a preference relation over several criteria. A common issue when defining a very precise and

F. Le Huédé (✉) · C. Labreuche · P. Savéant
THALES Research and Technology France,
domaine de Corbeville, 91401 Orsay cedex
e-mail: {fabien.lehuede; christophe.labreuche; pierre.saveant}@thalesgroup.com

M. Grabisch
Université Paris I Pantheon-Sorbonne

M. Grabisch
LIP 6, Université Pierre et Marie Curie (UPMC), 8, rue du Capitaine Scott 75015 Paris
e-mail: Michel.Grabisch@lip6.fr

elaborate preference relation, is elicitating the preferences of an expert in order to enable the framework's model to reproduce his choice in a combinatorial search space.

Multi-Criteria Decision Making (MCDM) proposes several methodologies and tools that provide accurate models for the preference of an expert over several criteria. Research in this area focuses mainly on constructing appropriate models for preferences and on processes that produce a correct elicitation of subjective preferences. MCDM models either evaluate or rank a given set of solutions (also called alternatives).

Surprisingly, few approaches integrate an MCDM model in their framework in order to benefit both of the capacities of the model for preference modeling and of the elicitation methodology that has been designed for this model. As a consequence, when a complex preference relation is needed, CP solvers offer few possibilities to handle the problem from the multi-criteria point of view.

1.1. Related work

Many solutions were proposed for integrating multi-criteria preferences in CP. A first category of approaches concentrates on finding a set of Pareto-optimal solutions for different objective functions (Gavanelli, 2002; Barichard and Hao, 2003; Ben Jaâfar et al., 2004). Alternatively, configuration problems and web applications have given rise to several approaches where a formalism allows the user to express partial preferences on criteria in a simple way (Junker, 2004; Pu and Faltings, 2004).

In this section, we focus on methods that allow a sufficiently precise description of the preference relation to be given in order to search for an (approximately) optimal solution. For this concern, most approaches rely on one of the two following frameworks.

The first well known framework is soft-constraints and in particular, Weighted CSPs (Bistarelli et al., 1999). It has been designed to solve over-constrained problems, but modeling multi-criteria problems using soft constraints is often suggested. Expressing preferences as constraints in order to describe a complete preference relation may be however problematic when the number of attributes to be taken into account is large. The method therefore needs a well founded and careful preference elicitation process. Indeed, even for limited partial preference relations, the simple example given in (Pu and Faltings, 2004) shows that one cannot manually modify weights in constraints and expect to keep a clear view of the relative importance of criteria in this relation.

The second framework, called CP-nets (Boutilier et al., 2004), proposes asking the user preference rules such as "if the main meal is fish, then I prefer white wine to red wine". A partial or complete preference relation can then be expressed using this kind of rules over the attributes of a solution. This framework is very expressive and can, in particular, handle context-dependent preferences. However, when the number of attributes and the number of values per attributes gets high, the description of a quasi-complete preference relation implies that a great number of rules are expressed. Recent studies propose integrating CP-nets in CP for optimization (Boutilier et al., 2004; Prestwich et al., 2005). (Prestwich et al., 2005) propose modeling CP-nets with hard constraints thanks to a new semantics. The use of implication constraints in this model suggests that the propagation of a CP-net in a combinatorial problem has few chances to be very efficient during the search for solutions. When the criteria of the problem are clearly different of the decision variables, it is likely that most solutions have to be constructed to ensure that the returned solution is not dominated.

Finally, (Kaymak and Sousa, 2003) proposes a framework for combining a multi-criteria model and a Fuzzy CSP. The constraints proposed in the following of this paper can be used to model the multi-criteria part in this framework.

1.2. Contributions of the paper

In this paper we study the integration of a model of the Multi-Attribute Utility Theory (MAUT) in CP. This model is introduced in Section 2. In Section 3, we propose a general scheme that offers a good flexibility for integrating elaborate multi-criteria models. We then introduce the principles of the propagation of the Aggregation constraint, which models the class of multi-criteria aggregation functions. The case of the Choquet integral is then studied in detail (Section 4). As this aggregation functions is particularly adapted to preference modeling, we apply the principles of Aggregation to define the Choquet constraint. Specific propagation algorithms are then designed and applied in Section 5 to a multi-criteria version of the examination timetabling problem.

2. Modeling multi-criteria preferences

In this paper, we focus on the *Multi-Attribute Utility Theory* framework (Keeney and Raiffa, 1976). This framework models the value of a solution through an overall evaluation, computed by an aggregation function according to its levels of satisfaction on a set of criteria. We introduce the properties of a multi-criteria aggregation function that are the most desirable in order to give a good representation of multi-criteria preferences. For this purpose the Choquet integral is a very general aggregation function which can model a wide range of decisional behaviors according to well founded elicitation processes (Grabisch and Roubens, 2000; Labreuche and Grabisch, 2003).

2.1. Preference modeling in Multi-Criteria Decision Making

Multi-Criteria Decision Making (MCDM) models subjective preferences in order to automate the determination of a preferred solution out of a set of alternatives. Hence, solving a typical multi-criteria decision problem consists in modeling the way an expert ranks a set of potential solutions, described by a set of *attributes* or *points of view*. To achieve this objective, the Multi-Attribute Utility Theory is mainly concerned with the construction of additive utility functions.

Let us denote by $\mathcal{N} = \{1, \ldots, n\}$ the set of criteria. We assume a set of *solutions* or *alternatives* $\mathcal{S}$ among which the decision maker must choose. Each solution is associated with a vector $a \in \Omega$ whose components $a_i \in \Omega_i$, $i \in \{1, \ldots, n\}$ represent the value of the solution for each point of view to be taken into account in the decision making process. A component a_i is called the *attribute* of a solution. Typically, in a multi-objective optimization context, each attribute would correspond to an objective function. According to these attributes, the modeling of the decision maker's preferences $\succeq$ is realized through an *overall utility function* $u : \Omega \to \mathbb{R}$ such that:

$$\forall a, b \in \Omega, a \succeq b \Leftrightarrow u(a) \geq u(b), \tag{1}$$

where $\succeq$ is a complete pre-order.

Classically, this overall evaluation function is split into two parts (Keeney and Raiffa, 1976):

– The *utility functions*, denoted $u_1(a_1), \ldots, u_n(a_n)$, map each attribute to a single satisfaction scale $\mathcal{E} \in \mathbb{R}$. They model the *performance* of a solution on the criteria and ensure

commensurateness between criteria, which is essential when several values have to be aggregated.
– The *aggregation function* aggregates the values returned by $u_1, \ldots, u_n$ and establishes the overall evaluation: $\forall a \in \Omega, u(a) = \mathcal{H}(u_1(a_1), \ldots, u_n(a_n))$.

where $u_i : \Omega_i \to \mathcal{E}$ and $\mathcal{H} : \mathcal{E}^n \to \mathcal{E}$. $u_i(a_i)$ is called the *utility* or *score* of the alternative a on the criterion i.

A common value for the satisfaction scale $\mathcal{E}$ is the [0, 1] interval. Establishing commensurateness between criteria allows us to work with comparable values. For example, this implies that we are able to express that a makespan of 20 days corresponds to the same level of satisfaction as a maximum tardiness of 3 days. Furthermore, when using compensatory aggregators (such as the weighted sum), it is important to work on an *interval scale*. This means that the difference between two values on the same criterion has to make sense. For example, it consists in deciding whether the difference of satisfaction of going from 5 days to 10 days is the same as going from 15 days to 20 days. To construct utility functions in the MAUT framework, we use the MACBETH methodology (Bana e Costa and Vansnick, 1994) (and its associated software), which is based on measurement theory. MACBETH asks the user to give some reference levels for a criterion and some general indications on the difference of satisfaction between values of this attribute.

2.2. Properties of a multi-criteria aggregation function

In decision aid, an aggregation function aggregates commensurate values which model criteria satisfaction levels. Some essential requirements have to be met by the aggregation function $\mathcal{H}$. Among these important properties, which are detailed in (Marichal, 1998), we denote:

– *Monotonicity* (M): $\mathcal{H}$ should be an increasing function. That is to say:

$$x_i > x_i' \Rightarrow \mathcal{H}(x_1, \ldots, x_i, \ldots, x_n) \geq \mathcal{H}(x_1, \ldots, x_i', \ldots, x_n).$$

Indeed, if one solution is better than another on at least one criterion and has equal performances on the other criteria, it cannot be of lesser quality than the other solution. This property is called *Strict Monotonicity* when the right side of the implication is a strict inequality.
– *Continuity* (C): $\mathcal{H}$ should be continuous with respect to its argument as the preferences of the expert generally evolve progressively.

In addition, $\mathcal{H}$ should satisfy (1) when $\succeq$ is defined. Consequently, if we want to use the same parametric model in several applications, the aggregation function has to be very flexible. Additive aggregation functions, usually used in MAUT, suppose *preferential independence* between criteria (Keeney and Raiffa, 1976), which is seldom the case in decision making. The aggregation function has to be able to model the importance of a criterion, but also interaction and compensation effects between criteria. As a result, we propose using an enhanced version of this theory, with a more general aggregation function.

2.3. The Choquet integral

In multi-criteria decision problems, some general aggregation functions such as the *Choquet integral* (Choquet, 1953; Grabisch, 1996) are often necessary in order to take into account not only the importance of each criterion, but also interaction phenomena between the criteria. In

order to generalize the weighted sum, (Sugeno, 1974) proposes assigning weights not only to each criterion separately, but also to any coalition of criteria. This weighting corresponds to a set function called "fuzzy measure".

Definition 1 (Fuzzy measure (Sugeno, 1974)). Let $\mathcal{P}(\mathcal{N})$ be the power set of $\mathcal{N}$. A fuzzy measure μ on $\mathcal{N}$is a function $\mu : \mathcal{P}(\mathcal{N}) \to [0, 1]$, satisfying the following axioms.

(i) $\mu(\emptyset) = 0,\ \mu(\mathcal{N}) = 1.$
(ii) $A \subset B \subset \mathcal{N}$ implies $\mu(A) \leq \mu(B)$.

Here, $\mu(A)$ represents the degree of importance of the subset of criteria $A \subset \mathcal{N}$. In the MCDM methodology, the fuzzy measure is used to model the decision maker preferences (Grabisch and Roubens, 2000). Then, the mono-dimensional utilities $u_1, \ldots, u_n$ are aggregated with the Choquet integral to produce the overall evaluation of an alternative.

Definition 2 (The Choquet integral (Choquet, 1953)). Let μ be a fuzzy measure on $\mathcal{N}$, and $u = (u_1, \ldots, u_n) \in [0, 1]^n$. The Choquet integral of u with respect to μ is defined by:

$$C_\mu(u_1, \ldots, u_n) = \sum_{i=1}^{n} u_{\sigma(i)}[\mu(A_{\sigma(i)}) - \mu(A_{\sigma(i+1)})], \tag{2}$$

where $\sigma(i)$ indicates a permutation on $\mathcal{N}$ such that $u_{\sigma(1)} \leq \cdots \leq u_{\sigma(n)}$, $A_{\sigma(i)} = \{\sigma(i), \ldots, \sigma(n)\}$ and $A_{\sigma(n+1)} = \emptyset$.

Thus, for three criteria, $C_\mu(u_1, u_2, u_3) = \mu(\{\sigma(3)\}) \times u_{\sigma(3)} + [\mu(\{\sigma(2), \sigma(3)\}) - \mu(\{\sigma(3)\})] \times u_{\sigma(2)} + [1 - \mu(\{\sigma(2), \sigma(3)\})] \times u_{\sigma(1)}$. The best score gets its individual weight whereas the others are corrected with respect to better criteria.

The Choquet integral is continuous, increasing, idempotent, stable for positive linear transformation and linear for a given order of its components. An axiomatization of its use for preference modeling has been introduced in Marichal (1998). Efficient indicators are also available for the semantic interpretation of the fuzzy measure (Grabisch, 2000). Ensuring that the Choquet integral is strictly increasing can be easily done setting $\forall A, B \subset \mathcal{N}, A \subset B \Rightarrow \mu(A) < \mu(B)$.

The Choquet integral is a general aggregation function which takes as particular cases the weighted sum, the min and the max. In addition, it allows every linear interpolations between these functions, which makes it very expressive.

Figure 1 shows two representations of the Choquet integral on two criteria. The first curve represents a case where the interaction between the two criteria is positive (they are said to be *complementary*). It models a preference relation where a solution has to be good on both criteria to be considered good. On the contrary, the right hand curve models *substitutive* criteria (i.e., negative interaction). In this case, a solution is considered good by the expert as soon as it is good on one criterion.

Efficient tools and methodologies have been created to establish a good multi-criteria model. In Thales, we use the Myriad© (Labreuche and Le Huédé, 2005) software to help the analyst in building the criteria hierarchy. It calls MACBETH to construct utility functions (§ 2.1), and determines the Choquet integral coefficients according to the decision maker preferences. As these coefficients are complex, they cannot be set by hand. The expert is asked to make pairwise comparisons on various solutions. They are used to build a linear program and deduce the fuzzy measure (Grabisch and Roubens, 2000).

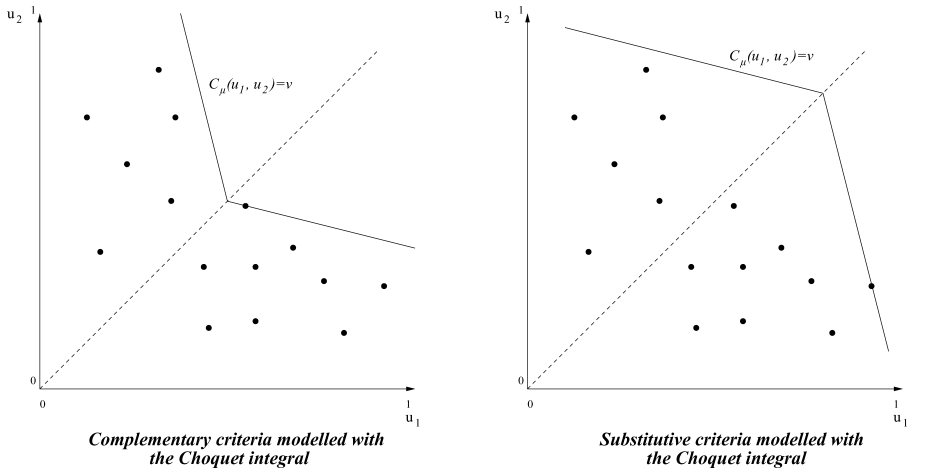

Fig. 1 Level curves of the Choquet integral for the aggregation of two criteria

Example 1 (The Choquet integral for students evaluation). Consider a director who would like to evaluate some students according to scores in mathematics (M), statistics (S) and languages (L). The director has some preferences such as *"for students good in mathematics, a student good in languages is preferred to one good in statistics"* and conversely *"for students bad in mathematics, a student good in statistics is preferred to one good in languages"*. Thus, for scores in [0, 1], this can be expressed as two "learning examples": $(0.8, 0.5, 0.4) \succ (0.8, 0.6, 0.3)$ and $(0.2, 0.6, 0.3) \succ (0.2, 0.5, 0.4)$. Then, to identify trade-offs between criteria, the director is asked to answer questions such as *"which value would you give to score α in the equivalence* $(0.5, 0.8, 0.4) \equiv (0.5, 0.7, \alpha)$*?"*. Setting $\alpha = 0.5$ and specifying that (M) is the most important criterion, we obtain the following fuzzy measure: $\mu(\{M\}) = \mu(\{S\}) = 0.5$, $\mu(\{L\}) = 0$, $\mu(\{M, S\}) = \mu(\{S, L\}) = 0.5$, $\mu(\{M, L\}) = 1$ that reflects the director requirements. The reader can note that, according to these preferences, the relative importance of S compared to L is conditional on M being good or bad. This type of decision strategy occurs quite often in preference modeling. They are represented for instance in TCP-nets in a qualitative framework. One can easily check that neither the weighted sum nor the minimum or the maximum can model these preferences.

For a more detailed application of this model in MCDM, the reader can report to (Labreuche and Le Huédé, 2005).

3. Integrating the MAUT model in CP

Our main objective is to capitalize on MCDM models and preferences elicitation methods in order to offer accurate multi-criteria optimization functionalities in CP. Considering that the MAUT model establishes an overall utility for each solution it can be quite naturally integrated in CP as a general objective function for multi-criteria optimization problems.

A simple MAUT model is composed of the following components (Section 2.1): *attributes*, which are generally objective functions in optimization, *utility functions* (one per attribute), which establish the scores of a solution on the criteria, *an aggregation function*, that makes the synthesis of the scores in order to calculate the overall evaluation of a solution. More elaborate preference models generally include several hierarchical levels of aggregation.

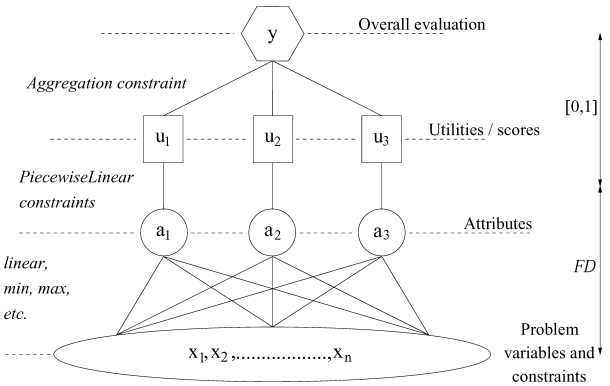

Fig. 2 MAUT integration scheme

For an efficient and flexible integration of MAUT in CP we propose modeling each function with a dedicated global constraint, considering more particularly the Choquet integral case. This leads us to the following integration scheme.

3.1. Integration scheme of the MAUT model

Let us consider a simple model with three criteria and a multi-criteria aggregation function $\mathcal{H}$. Figure 2 describes the proposed integration approach, where each function of the preference model is modeled by a global constraint.

According to this scheme, modeling a multi-criteria optimization problem implies defining several kind of variables: $y \in [0, 1]$ corresponds to the overall evaluation, $u_1, \ldots, u_n \in [0, 1]$ are the scores of a solution over each criterion, $a_1, \ldots, a_n$ are finite domain variables that model the attributes of the problem, $x_1, \ldots, x_m$ are the finite domain variables of the combinatorial problem.

The combinatorial problem is modeled with constraints on variables $x_1, \ldots, x_m$ that are connected to variables $a_1, \ldots, a_n$ by objective functions using constraints such as sum, min or max. The two upper levels of the scheme represent the multi-criteria model. Each attribute $a_i, i \in \{1, \ldots, n\}$ is connected to a score u_i by a PiecewiseLinear constraint that models a utility function. In order to enforce the relation $y = \mathcal{H}(u_1, \ldots, u_n)$, we introduce the Aggregation constraint which connects $u_1, \ldots, u_n$ to the variable $y \in [0, 1]$ that will be maximized.

Hence, two global constraints are needed: PiecewiseLinear models piecewise linear functions and Aggregation models the aggregation function $\mathcal{H}$.

In this scheme, Aggregation represents the generic class of constraints that model MCDM aggregation functions (§ 2.2). Some general properties for the propagation of this class of constraints are introduced in Section 3.2. These properties are then applied in Section 4 to the Choquet integral for the design of the Choquet constraint, which can then be seen as a specialization of Aggregation.

Remark 1 (Discretizing the continuous part of the model). In this integration scheme, variables $x_1, \ldots, x_m$, $a_1, \ldots, a_n$ are finite domain variables and $y, u_1, \ldots, u_n$ belong to the $[0, 1] \subset \mathbb{R}$ interval. To integrate this scheme in a finite domain solver, we discretized the $[0, 1]$ interval into the $\{0, \ldots, 10^6\}$ domain. As for propagation on intervals in CP (Lhomme, 1993), this approach needs "outward rounding" rules (Michel et al., 2001) to guarantee the correctness of algorithms. In particular, using outward rounding on discretized variables does not perturb the results presented in this paper. An interval $[\underline{x}, \overline{x}] \subset [0, 1]$ is represented by

the $\{\lfloor 10^6 \times \underline{x} \rfloor, \ldots, \lceil 10^6 \times \overline{x} \rceil\}$ domain. Discretizing [0, 1] and outward rounding are often considered as implicit in the following of this paper.

Remark 2 (The PiecewiseLinear constraint). Continuous piecewise linear functions are not imposed by MAUT but they correspond to the functions that are designed by the MACBETH software (Bana e Costa and Vansnick, 1994) (§ 2.1). As no a priori shape can be assumed for utility functions, most required shapes can be approximated with piecewise linear functions, which also offer a good understanding of their behavior between two characteristic points.

The design of the PiecewiseLinear constraint has been proposed by P. Refalo using linear relaxations (Refalo, 1999). Here, we consider only strict monotonous utility functions which are very easy to integrate in CP. Hence, to propagate $y = f(x)$ where f is a strictly increasing piecewise linear function, we just have to maintain $\underline{y} \geq \lfloor f(\underline{x}) \rfloor$ and $\overline{y} \leq \lceil f(\overline{x}) \rceil$ for the "discretized" variable y, and $\underline{x} \geq \lceil f^{-1}(\underline{y}) \rceil$ and $\overline{x} \leq \lfloor f^{-1}(\overline{y}) \rfloor$ for the integer variable x.

3.2. The Aggregation constraint

As stated in § 2.2, multi-criteria aggregation functions generally verify the Strict Monotonicity and Continuity properties. Some common principles can be identified for propagating the constraints that model these functions. To introduce these principles we define the Aggregation constraint, which establishes and propagates the equality between a variable y and the aggregation of scores $u_1, \ldots, u_n$ by a function $\mathcal{H}$. Mathematically we want to enforce $y = \mathcal{H}(u_1, \ldots, u_n)$.

Definition 3 (The Aggregation *constraint).* Let $\mathcal{N}$ be a set of n criteria, let $\{y\} \bigcup \{u_1, \ldots, u_n\}$ be a set of variables ranging over [0, 1] and let $\mathcal{H}$ be a continuous and strictly increasing multi-criteria aggregation function. The Aggregation constraint enforces the relation $y = \mathcal{H}(u_1, \ldots, u_n)$ and is denoted aggregation$(\mathcal{H}, y, \{u_1, \ldots, u_n\})$.

According to $\mathcal{H}$, the Aggregation constraint can be specialized to an adequate constraint that inherits its properties. Many studies deal with the modeling of the weighted sum in CP (see e.g. (Apt, 1998; Zhang and Yap, 2000)). Particularly, Apt (Apt, 1998) defines propagation rules for the Linear Equality constraint on $\mathbb{R}$. These results can be generalized to the whole set of multi-criteria aggregation functions.

3.3. Arc-B-Consistency of the Aggregation constraint

As the MAUT model relies on real variables, variables domains are limited to intervals throughout of the paper. We denote $[\underline{x}, \overline{x}]$ the domain of a variable x. Hence, the propagation of *Aggregation* can be achieved by maintaining the arc-B-consistency for this constraint.

Definition 4 (Arc-B-consistency). (Lhomme, 1993) Given a constraint c over q variables $x_1, \ldots, x_q$, and a domain $d_i = [\underline{x_i}, \overline{x_i}]$ for each variable x_i, c is said to be "arc-B-consistent" if and only if for any variable x_i and for each bound $v_i = \underline{x_i}$ and $v_i = \overline{x_i}$, there exist values $v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_q$ in $d_1, \ldots, d_{i-1}, d_{i+1}, \ldots, d_q$ such that $c(v_1, \ldots, v_q)$ holds.

Arc-B-consistency is weaker than the *arc-consistency* property. Indeed a constraint is arc-consistent when, for each value in the domain of each of the constraint's variables, there is a set of values in the domain of the other variables that verifies the constraint.

The Strict Monotonicity and Continuity properties of function $\mathcal{H}$ enables us to define four necessary and sufficient conditions that ensure Arc-B-consistency with respect to the Aggregation constraint and avoid searching for a set of value that verify the constraint for each variable bound.

Proposition 1. *(Arc-B-consistency with respect to the Aggregation constraint) Let $\mathcal{H}$ be a strictly increasing and continuous aggregation function and C =Aggregation($\mathcal{H}$, y, $\{u_1, \ldots, u_n\}$) be an Aggregation constraint. C is Arc-B-consistent if and only if the following four conditions hold:*

(AB1) $\underline{y} \geq \mathcal{H}(\underline{u_1}, \ldots, \underline{u_n})$
(AB2) $\overline{\underline{y}} \leq \mathcal{H}(\overline{\underline{u_1}}, \ldots, \overline{\underline{u_n}})$
(AB3) $\forall k \in \{1, \ldots, n\} : \mathcal{H}(\overline{u_1}, \ldots, \overline{u_{k-1}}, \underline{u_k}, \overline{u_{k+1}}, \ldots, \overline{u_n}) \geq \underline{y}$
(AB4) $\forall k \in \{1, \ldots, n\} : \mathcal{H}(\underline{u_1}, \ldots, \underline{u_{k-1}}, \overline{\underline{u_k}}, \underline{u_{k+1}}, \ldots, \underline{u_n}) \leq \overline{\underline{y}}$

Notation 1. For the sake of concision, we use the following notations: $\mathcal{H}(\underline{u}) = \mathcal{H}(\underline{u_1}, \ldots, \underline{u_n})$, $\mathcal{H}(\overline{u}) = \mathcal{H}(\overline{u_1}, \ldots, \overline{u_n})$, $\mathcal{H}(\overline{u_{-k}}, \underline{u_k}) = \mathcal{H}(\overline{u_1}, \ldots, \overline{u_{k-1}}, \underline{u_k}, \overline{u_{k+1}}, \ldots, \overline{u_n})$, $\mathcal{H}(\underline{u_{-k}}, \overline{u_k}) = \mathcal{H}(\underline{u_1}, \ldots, \underline{u_{k-1}}, \overline{u_k}, \underline{u_{k+1}}, \ldots, \underline{u_n})$.

Proof: Let us consider the y variable and assume conditions (AB1) and (AB2) are verified. Since since $\mathcal{H}$ is increasing, (AB1) and (AB2) implies $\underline{y}$ and $\overline{y}$ belong to the interval $[\mathcal{H}(\underline{u}), \mathcal{H}(\overline{u})]$. Function $\mathcal{H}$ being continuous, for each value $v \in \{\underline{y}, \overline{y}\}$, there exist values $v_1, \ldots, v_n$ in $[\underline{u_1}, \overline{u_1}], \ldots, [\underline{u_n}, \overline{u_n}]$ such that $v = \mathcal{H}(v_1, \ldots, v_n)$. Conversely, for any value $v \notin [\mathcal{H}(\underline{u}), \mathcal{H}(\overline{u})]$, $\mathcal{H}(\underline{u})$ and $\mathcal{H}(\overline{u})$ being the lowest and highest values that can be taken by $\mathcal{H}$ respectively, there is no set of values that satisfies C in the domains of $u_1, \ldots, u_n$. The domain $[\underline{y}, \overline{y}]$ is arc-B-consistent w.r.t. the Aggregation constraint C if and only if conditions (AB1) and (AB2) are verified.

Similarly, for a variable $u_k \in \{u_1, \ldots, u_n\}$, (AB3) and (AB4) ensure that $[\underline{u_k}, \overline{u_k}]$ is arc-B-consistent with respect to C. □

Proposition 1 is very straightforward. It should be noted that it is not valid however when variables are integer or when $\mathcal{H}$ is not continuous.

Conditions (AB1) and (AB2) propagate Aggregation on the domain of variable y and for each variable u_k, $k \in \{1, \ldots, n\}$, conditions (AB3) and (AB4) propagate on $\underline{u_k}$ and $\overline{u_k}$ respectively. A first remark is that when (AB1) or (AB2) is not verified, it is easy to compute a new valid domain for y: $[max(\underline{y}, \mathcal{H}(\underline{u})), min(\overline{y}, \mathcal{H}(\overline{u}))]$. This deduction may be more difficult for a variable $u_k \in \{u_1, \ldots, u_n\}$. It implies expressing u_k with respect to the other variables, which may be complicated if the inverse function of $\mathcal{H}$ is not easily calculable.

In the following sections we show that these four conditions ensure that the Aggregation constraint is arc-consistent (§ 3.4) and that a propagation algorithm needs to enforce each condition only once to ensure the consistency of the whole constraint. The propagation algorithm is said to be *idempotent* (§ 3.5).

3.4. Arc-consistency of the Aggregation constraint

For some constraints, when the domain of the variables are restricted to intervals enforcing arc-B-consistency allows to ensure that it is also arc-consistent (Lhomme, 1993). As for the Linear Equality constraint (Apt, 1998), this property is verified for Aggregation.

Theorem 1 (Arc-consistency with respect to the Aggregation constraint). *Let* $\{y\} \bigcup \{u_1, \ldots, u_n\}$ *be a set of variables defined on the* $[0, 1]$ *interval, let* $\mathcal{H}$ *be a strictly increasing continuous aggregation function and* $C =$*Aggregation*$(\mathcal{H}, y, \{u_1, \ldots, u_n\})$ *be an Aggregation constraint.* C *is arc-consistent if and only if* c *is arc-B-consistent.*

Proof: Function $\mathcal{H}$ being continuous, considering conditions (AB1) and (AB2) and according to the intermediate values theorem, for all value $v \in [\underline{y}, \overline{y}]$, there exists a set of values $v_1, \ldots, v_n$ in $[\underline{u_1}, \overline{u_1}], \ldots, [\underline{u_n}, \overline{u_n}]$ such that $v = \mathcal{H}(v_1, \ldots, v_n)$. Similarly, considering (AB3) and (AB4), for all set of values $v_k \in [\underline{u_k}, \overline{u_k}]$, we have $\mathcal{H}(\underline{u_{-k}}, v_k) \geq \underline{y}$. There is therefore a set of values $v, v_1, \ldots, v_{k-1}, v_{k+1}, \ldots, v_n$ in $[\underline{y}, \overline{y}], [\underline{u_1}, \overline{u_1}], \ldots, [\underline{u_{k-1}}, \overline{u_{k-1}}], [\underline{u_{k+1}}, \overline{u_{k+1}}], \ldots, [\underline{u_n}, \overline{u_n}]$ such that $v = \mathcal{H}(v_1, \ldots, v_n)$. □

Again,this theorem may seem trivial since variable domains are limited to intervals. However, it may be violated for non-monotonic functions such as the absolute value of a real number.

3.5. Idempotency of the Aggregation constraint

Maintaining the consistency of the Aggregation constraint implies continuously checking the conditions of Proposition 1 and reducing the variables domains to keep them verified during the search. Hence, virtually any event that reduces the variables domains may trigger the filtering algorithm.

An interesting property is that a single calculation of all bound reductions generated by the consistency conditions of Aggregation is enough to maintain its arc-consistency. Without any event that is external to the constraint, calculating all bounds a second time would not reduce any domain. The constraint is said to be *idempotent* (Apt, 1999). To express this property, the propagation mechanism is often defined as a domain reduction function (Apt, 1999) which, from a set of domains $\{d_1, \ldots, d_m\}$ returns a set of domains $\{d'_1, \ldots, d'_m\}$ deduced by the propagation.

Theorem 2. *(Idempotency of the* Aggregation *constraint) Let* $y, u_1, \ldots, u_n$ *be a set of variables defined on domains* $d_y, d_{u_1}, \ldots, d_{u_n}$. *Let* $\mathcal{F}_{Agg}$ *a domain reduction function that propagates the arc-B-consistency conditions of an* Aggregation *constraint on variables* $y, u_1, \ldots, u_n$. *Function* $\mathcal{F}_{Agg}$ *is idempotent. That is to say:* $\mathcal{F}_{Agg}(\mathcal{F}_{Agg}(d_y, d_{u_1}, \ldots, d_{u_n})) = \mathcal{F}_{Agg}(d_y, d_{u_1}, \ldots, d_{u_n})$. *By extension, the* Aggregation *constraint is said to be idempotent.*

Proof: Aggregation is idempotent iff propagating one arc-B-consistency condition cannot provoke the violation of another. A bound reduction can modify the validity of a condition if it appears in this condition. Let us consider the propagations that can be deduced from Proposition 1:

Increasing of $\underline{y}$ *due to the propagation of (AB1):* $\underline{y}$ appears in (AB3). The propagation being $\underline{y} = \mathcal{H}(\underline{u}), \forall k \in \{1, \ldots, n\}$, $\mathcal{H}(\overline{u_{-k}}, \underline{u_k}) \geq \mathcal{H}(\underline{u}) = \underline{y}$. Condition (AB3) is true for all

$k \in \{1, \ldots, n\}$. Similarly, the propagation of (AB2) does not perturb the validity of other conditions.

Increasing of $\underline{u_i}$ due to the propagation of (AB3): $\underline{u_i}$ appears in (AB1) and (AB4) for all score u_k, $k \neq i$. Propagating (AB3) for $k = i$ ensure $\underline{y} = \mathcal{H}(\overline{u_{-i}}, \underline{u_i}) \geq \mathcal{H}(\underline{u})$, which validates (AB1). In addition, $\forall k \in \{1, \ldots, n\}, k \neq i$, $\overline{y} \geq \underline{y} = \mathcal{H}(\overline{u_{-i}}, \underline{u_i}) \geq \mathcal{H}(\underline{u_{-k}}, \overline{u_k})$. Condition (AB4) is therefore also true. Similarly, the propagation of (AB4) does not perturb the validity of other conditions.

As a results, checking each of the $2n + 2$ conditions once in any order ensures that all conditions are true. □

Hence, a basic propagation algorithm for an Aggregation constraint checks and propagates each of the $2n + 2$ conditions of Proposition 1 every time an external event modifies a bound of its variables during the search. CP solvers such as Eclair© (Museux et al., 2003) generally trigger a suitable propagation algorithm according to the nature of the event and to the variable on which it happens. The propagation is then said to be *incremental*. Such algorithms have been extensively described in Le Huédé (2003) (in French) for the Aggregation constraint.

4. Application to the propagation of the Choquet integral

In this section we apply the propagation principles identified for a general multicriteria aggregation function $\mathcal{H}$ to the Choquet integral. According to the integration scheme proposed in Section 3, we propose specializing Aggregation to the Choquet constraint.

4.1. The Choquet constraint

Considering n variables $u_1, \ldots, u_n \in [0, 1]$ and a variable $y \in [0, 1]$, our aim is to establish and propagate the relation $y = C_\mu(u_1, \ldots, u_n)$ for a given fuzzy measure μ. Hence, we want to enforce $y = \sum_{i=1}^{n} u_{\sigma(i)}[\mu(A_{\sigma(i)}) - \mu(A_{\sigma(i+1)})]$, where $\sigma(i)$ indicate a permutation on $\mathcal{N}$ such that $u_{\sigma(1)} \leq \cdots \leq u_{\sigma(n)}$, $A_{\sigma(i)} = \{\sigma(i), \ldots, \sigma(n)\}$ and $A_{\sigma(n+1)} = \emptyset$. To maintain this relation, we propose defining the Choquet constraint:

Definition 5 (The Choquet *constraint).* Let $\mathcal{N}$ be a set of n criteria, let $\{y\} \bigcup \{u_1, \ldots, u_n\}$ be a set of variables ranging over [0, 1] and let $\mathcal{M} = \{\mu(\emptyset), \mu(\{1\}), \mu(\{2\}), \ldots, \mu(\{1, \ldots, n\})\}$ be the values of a fuzzy measure μ for each element of $\mathcal{P}(\mathcal{N})$. The Choquet constraint enforces the relation $y = C_\mu(u_1, \ldots, u_n)$ and is denoted choquet$(y, \{u_1, \ldots, u_n\}, \mathcal{M})$.

4.2. Propagating the Choquet constraint

The Choquet integral being continuous and strictly increasing (providing adequate conditions on μ (§ 2.3)), the Choquet constraint belongs to the Aggregation constraint class. As a result, maintaining the arc-B-consistency w.r.t. Choquet can be done by checking the consistency conditions of Proposition 1 with $\mathcal{H} = C_\mu$.

As previously observed in Section 3.3, enforcing conditions (AB1) and (AB2) is equivalent to assigning the domain $[max(\underline{y}, C_\mu(\underline{u})), min(\overline{y}, C_\mu(\overline{u}))]$ to variable y. Expressing a score u_k with respect to the other variables in order to compute its new domain is however more complicated. This implies deducing a function $\mathcal{F}$ from equation $y = C_\mu(u_1, \ldots, u_n)$ such that $u_k = \mathcal{F}(y, u_1, \ldots, u_{k-1}, u_{k+1}, \ldots, u_n)$ without knowing the respective order of the scores.

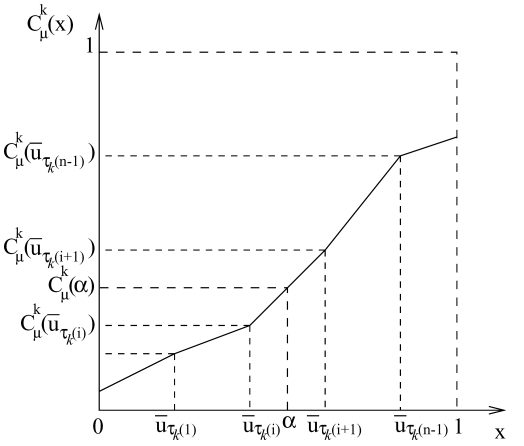

Fig. 3 Graphical representation of $C_\mu^k(x)$

4.3. Calculating the lower bound of a score variable

Assuming condition (AB3) is violated for a given score $u_k, k \in \{1, \ldots, n\}$. We have $C_\mu(\overline{u_1}, \ldots, \overline{u_{k-1}}, \underline{u_k}, \overline{u_{k+1}}, \ldots, \overline{u_n}) < \underline{y}$. To restore the validity of this condition, a new value denoted $\breve{u}_k$ must be found for the lower bound $\underline{u_k}$ such that $C_\mu(\overline{u_{-k}}, \breve{u}_k) = \underline{y}$. In this equation, the upper bounds $\overline{u_1}, \ldots, \overline{u_{k-1}}, \overline{u_{k+1}}, \ldots, \overline{u_n}$ are actually known, so is their relative order. The only unknown is therefore the relative order of $\breve{u}_k$ with respect to these bounds.

In the following paragraph we propose defining algorithm **LB** (i) that first determines the relative order of $\breve{u}_k$ with respect to the other bounds and finally returns its value.

Description of function $C_\mu^k(x) = C_\mu(\overline{u_{-k}}, x)$

Notation 2. Let us denote $C_\mu^k : [0, 1] \to [0, 1]$ the function such that $C_\mu^k(x) = C_\mu(\overline{u_{-k}}, x)$. Let τ_k be a permutation on indices $1, \ldots, k-1, k+1, \ldots, n$ such that $\overline{u_{\tau_k(1)}} \leq \cdots \leq \overline{u_{\tau_k(n-1)}}$ and let us denote $\overline{u_{\tau_k(0)}} = 0$ and $\overline{u_{\tau_k(n)}} = 1$. In addition we define the set $A_{\tau_k(i)}^{-k} = \{\tau_k(i), \ldots, \tau_k(n-1)\}$ and we denote $A_{\tau_k(n)}^{-k} = \emptyset$.

Figure 3 shows a possible representation of C_μ^k. It is an increasing piecewise linear function whose vertices correspond to the points where x is equal to $0, \overline{u_{\tau_k(1)}}, \ldots, \overline{u_{\tau_k(n-1)}}, 1$.

Consider a real value $\alpha \in [0, 1]$. We denote $i \in \{0, \ldots, n-1\}$ the index that characterizes the segment on which the point $(\alpha, C_\mu^k(\alpha))$ is located, that is to say, $\alpha \in [\overline{u_{\tau_k(i)}}, \overline{u_{\tau_k(i+1)}}]$. According to the definition of the Choquet integral, function $C_\mu^k(\alpha)$ can be written as follows: $\forall \alpha \in [0, 1], \exists i \in \{0, \ldots, n-1\}$ such that $\alpha \in [\overline{u_{\tau_k(i)}}, \overline{u_{\tau_k(i+1)}}]$:

$$\begin{aligned} C_\mu^k(\alpha) = & \sum_{j=1}^{i} \overline{u_{\tau_k(j)}} \times \left[\mu\left(A_{\tau_k(j)}^{-k} \cup \{k\}\right) - \mu\left(A_{\tau_k(j+1)}^{-k} \cup \{k\}\right)\right] \\ & + \alpha \times \left[\mu(A_{\tau_k(i+1)}^{-k} \cup \{k\}) - \mu\left(A_{\tau_k(i+1)}^{-k}\right)\right] \\ & + \sum_{j=i+2}^{n} \overline{u_{\tau_k(j-1)}} \times \left[\mu\left(A_{\tau_k(j-1)}^{-k}\right) - \mu\left(A_{\tau_k(j)}^{-k}\right)\right] \end{aligned} \tag{3}$$

Hence, to calculate $\check{u}_k$ such that $C_\mu^k(\check{u}_k) = \underline{y}$, we first have to determine on which linear part of the curve the point $(\check{u}_k, \underline{y})$ is located. This is equivalent to finding the index $i^\star$ such that $C_\mu^k(\overline{u_{\tau_k(i^\star)}}) < \underline{y} \leq C_\mu^k(\overline{u_{\tau_k(i^\star+1)}})$. Calculating $\check{u}_k$ can then be done with respect to point $(\overline{u_{\tau_k(i^\star)}}, C_\mu^k(\overline{u_{\tau_k(i^\star)}}))$ and to the slope of this linear part.

Locating lower bound $\check{u}_k$

As stated before, we search for $i^\star \in \{0, \ldots, n-1\}$, such that: $C_\mu^k(\overline{u_{\tau_k(i^\star)}}) < \underline{y} \leq C_\mu^k(\overline{u_{\tau_k(i^\star+1)}})$.

A first remark is that $\check{u}_k \geq \underline{u_k}$ and $\check{u}_k \leq \overline{u_k}$ ($C_\mu^k(\overline{u_k}) \geq \underline{y}$). In addition, $\forall k \in \{1, \ldots, n\}$, $C_\mu(\overline{u}) = C_\mu^k(\overline{u_k})$. Considering that the lower bounds of all scores will be calculated successively, we propose to explore function C_μ^k from edge to edge, starting from the already calculated point $(\overline{u_k}, C_\mu(\overline{u}))$ (which belongs to C_μ^k for all $k \in \{1, \ldots, n\}$), until $i^\star$ is found.

In order to avoid multiple calculations of the Choquet integral during this exploration, we use the following transition formula that allows going from a point with index i to a point with index $i-1$:

Proposition 2. *(Transition formula for function C_μ^k) Consider an index $i \in \{1, \ldots, n\}$. The value of C_μ^k at $\overline{u_{\tau_k(i-1)}}$ can be calculated with respect to $\overline{u_{\tau_k(i)}}$ and $C_\mu^k(\overline{u_{\tau_k(i)}})$ according to the following transition formula:*

$$C_\mu^k(\overline{u_{\tau_k(i-1)}}) = C_\mu^k(\overline{u_{\tau_k(i)}}) - (\overline{u_{\tau_k(i)}} - \overline{u_{\tau_k(i-1)}})\big(\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i)}^{-k}\big)\big) \tag{4}$$

Proof: If we express $C_\mu^k(\overline{u_{\tau_k(i)}})$ and $C_\mu^k(\overline{u_{\tau_k(i-1)}})$ according to Equation (3) and remove common terms, we have:

$$\begin{aligned}
C_\mu^k(\overline{u_{\tau_k(i)}}) - C_\mu^k(\overline{u_{\tau_k(i-1)}}) &= \overline{u_{\tau_k(i)}}\big[\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i+1)}^{-k} \cup \{k\}\big)\\
&\quad + \overline{u_{\tau_k(i)}}\big[\mu\big(A_{\tau_k(i+1)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i+1)}^{-k}\big)\big]\\
&\quad - \overline{u_{\tau_k(i-1)}}\big[\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i)}^{-k}\big)\big]\\
&\quad - \overline{u_{\tau_k(i)}}\big[\mu\big(A_{\tau_k(i)}^{-k}\big) - \mu\big(A_{\tau_k(i+1)}^{-k}\big)\big]\\
&= \overline{u_{\tau_k(i)}}\big[\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i)}^{-k}\big)\big]\\
&\quad - \overline{u_{\tau_k(i-1)}}\big[\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i)}^{-k}\big)\big]\\
&= (\overline{u_{\tau_k(i)}} - \overline{u_{\tau_k(i-1)}}) \times \big[\mu\big(A_{\tau_k(i)}^{-k} \cup \{k\}\big) - \mu\big(A_{\tau_k(i)}^{-k}\big)\big]
\end{aligned}$$

□

Remark 3. Let τ be a permutation of indices $1, \ldots, n$ such that $\overline{u_{\tau(1)}} \leq \cdots \leq \overline{u_{\tau(n)}}$. Let $r_k = \tau^{-1}(k)$ be the rank of $\overline{u_k}$ in the sorted set of scores upper bounds. Then we have $\overline{u_k} \in \big[\overline{u_{\tau_k(r_k-1)}}, \overline{u_{\tau_k(r_k)}}\big]$. If the algorithm starts the exploration of $C_\mu^k(x)$ from $(\overline{u_k}, C_\mu(\overline{u}))$, according to (4), the first edge below $(\overline{u_k}, C_\mu(\overline{u}))$ on the curve of function $C_\mu^k(x)$ takes the

following value on the y-axis:

$$C_\mu^k(\overline{u_{\tau_k(r_k-1)}}) = C_\mu(\overline{u}) - (\overline{u_k} - \overline{u_{\tau_k(r_k-1)}}) \times \left[\mu(A^{-k}_{\tau_k(r_k)} \cup \{k\}) - \mu(A^{-k}_{\tau_k(r_k)})\right]$$

In addition, for all $j \in \{1, \ldots, r_k - 1\}$, $\tau_k(j) = \tau(j)$. Knowing $\breve{u}_k \leq \overline{u_k}$, in a more general propagation algorithm, permutation τ can be used for all lower bound calculations instead of the n permutations $\tau_1, \ldots, \tau_n$.

Calculating lower bound $\breve{u}_k$

Finally, determining $i^\star$ allows us to deduce $\breve{u}_k$:

$$\breve{u}_k = \overline{u_{\tau_k(i^\star)}} + \frac{\underline{y} - C_\mu^k(\overline{u_{\tau_k(i^\star)}})}{\mu\left(A^{-k}_{\tau_k(i^\star+1)} \cup \{k\}\right) - \mu\left(A^{-k}_{\tau_k(i^\star+1)}\right)} \tag{5}$$

Proof: Since values $\breve{u}_k$ and $\overline{u_{\tau_k(i^\star)}}$ both belong to $[\overline{u_{\tau_k(i^\star)}}, \overline{u_{\tau_k(i^\star+1)}}]$, according to (4):

$$C_\mu^k(\breve{u}_k) - C_\mu^k(\overline{u_{\tau_k(i^\star)}}) = (\breve{u}_k - \overline{u_{\tau_k(i^\star)}}) \times \left[\mu\left(A^{-k}_{\tau_k(i^\star+1)} \cup \{k\}\right) - \mu\left(A^{-k}_{\tau_k(i^\star+1)}\right)\right]$$

Replacing $C_\mu^k(\breve{u}_k)$ by $\underline{y}$ in this equation results in equation (5). As the fuzzy measure is assumed to be strictly monotonic, no division by zero can be obtained. □

Algorithm

In the previous paragraph we detailed the steps of the determination of the lower bound of a score u_k, $k \in \{1, \ldots, n\}$ in order to maintain the arc-B-consistency condition (AB3) of Proposition 1. Figure 4 describes the **LB**(k) algorithm, which implements the proposed approach (a similar algorithm **UB**(k) is used to compute the upper bound of u_k).

The main loop of this algorithm locates the lower bound $\breve{u}_k$. It applies the transition formula (4) by decreasing variable $i^\star$ until the reached point is either lower or equal to $\underline{y}$ on the y-axis or lower to $\underline{u_k}$ on the x-axis.

Finally, when $\breve{u}_k \in [\overline{u_{\tau(i^\star)}}, \overline{u_{\tau(i^\star+1)}}]$ with $\overline{u_{\tau(i^\star+1)}} < \underline{u_k}$, the algorithm returns 0 and does not trigger any propagation. Otherwise, the value $\breve{u}_k$ is calculated according to (5).

The main loop in this algorithm represents at most a set of n constant time operations. The complexity of this propagation algorithm is therefore $O(n)$. As a result, propagating every conditions of the constraint has a complexity of $O(n^2)$.

4.4. An example of propagations generated by the Choquet constraint

Let y, u_1, u_2, u_3 be four variables and let $\mathcal{M} = \{\mu_0, \mu_1, \ldots, \mu_{123}\}$ be a fuzzy measure such that: $y \in [0.4, 1]$, $u_1 \in [0, 0.2]$, $u_2 \in [0, 0.8]$, $u_3 \in [0, 0.2]$ and $\mu_0 = 0$, $\mu_1 = 0.1$, $\mu_2 = 0.4$, $\mu_3 = 0.1$, $\mu_{12} = 0.5$, $\mu_{13} = 0.2$, $\mu_{23} = 0.6$, $\mu_{123} = 1$. If we set the constraint choquet$(y, \{u_1, u_2, u_3\}, \mathcal{M})$, we obtain the following propagations:

On the y variable, $C_\mu(0, 0, 0) = 0$ and $C_\mu(0.2, 0.8, 0.2) = 0.2 \times (1 - 0.6) + 0.2 \times (0.6 - 0.4) + 0.8 \times 0.4 = 0.44$. Therefore from (AB2) (Proposition 1) we can conclude that $y \in [0.4, 0.44]$.

Fig. 4 Calculation algorithm for the lower bound of a score u_k

```
LB(k) : float
    i* ← τ^{-1}(k)
    u_{i*} ← ū_k
    ŭ_k ← 0
    Cu_{i*} ← C_μ(ū)
    A^{-k}_{τ(i*)} ← {τ(i*+1), ..., τ(n)}
    while ((Cu_{i*} > y)&(u_{i*} > u_k))
        p ← μ(A^{-k}_{τ(i*)} ∪ {k}) − μ(A^{-k}_{τ(i*)})
        Cu_{i*} ← Cu_{i*} − p × (u_{i*} − u_{τ(i*−1)})
        u_{i*} ← u_{τ(i*−1)}
        A^{-k}_{τ(i*)} ← A^{-k}_{τ(i*)} ∪ τ(i* − 1)
        i* ← i* − 1
    endwhile
    if (Cu_{i*} = y)
        ŭ_k ← u_{i*}
    else if (Cu_{i*} < y)
            ŭ_k ← u_{i*} + (y − Cu_{i*}) / p
        endif
    endif
    return ŭ_k
end
```

On score u_1: Let $\breve{u}_1$ be the lower bound of u_1 that can be deduced from (AB3), i.e., such that $C_\mu(\breve{u}_1, 0.8, 0.2) = 0.4$. We have already calculated $C_\mu(0.2, 0.8, 0.2) = 0.44$. Therefore we can conclude that $\breve{u}_1 \in [0, 0.2)$ and from Equation (5): $\breve{u}_1 = \frac{0.4 - C_\mu(0,0.8,0.2)}{\mu_{123} - \mu_{23}} = \frac{0.4-0.36}{0.4} = 0.1$. Considering the calculation of an upper bound for u_1, we can notice that we have reduced $\overline{y}$ such that $\overline{y} = C_\mu(\overline{u_1}, \overline{u_2}, \overline{u_3})$. It follows that $\overline{y} \geq C_\mu(\overline{u_1}, \underline{u_2}, \underline{u_3})$. Therefore condition (AB4) is verified and we can conclude that $\overline{u_1}$ will not be reduced.

If we follow the same reasoning for u_2 and u_3, we finally obtain: $u_1 \in [0.1, 0.2]$, $u_2 \in [0.7, 0.8]$ and $u_3 \in [0.12, 0.2]$.

5. Experimentation

The propagation of the Choquet constraint has been evaluated on small instances of the multi-criteria examination timetabling problem.

5.1. The examination timetabling problem

Given a set of examinations, a set of students each enrolled for a given list of examinations, a set of rooms of fixed capacities and a set of periods, the examination timetabling problem consists in assigning a period and a room to each examination such that (i) two examinations that are given to a same student cannot be planned on the same period and (ii) the capacity of a room cannot be exceeded. We assume that as long as (i) and (ii) hold, several examinations can take place in the same room at the same time but that the number of students attending an examination cannot be distributed over several rooms.

A simple multi-criteria model has been constructed based on three attributes: the date of the last examination planned (criterion *duration of the examination*), the number of rooms used (criterion *rooms employment*) and the number of times a student has two consecutive examinations (criterion *spreading of the exams*). These criteria are aggregated using the Choquet integral (the whole multi-criteria model is more precisely described in Le Huédé (2003)).

5.2. Instances

Small scenarios have been constructed in order to evaluate the performance of the propagation in complete search. The main characteristics of these scenarios are described in the following table:

	Number of periods	Number of exams	Number of rooms	Number of students
Sc. 12	9	12	2	49
Sc. 15	9	15	3	56
Sc. 20	11	20	2	104
Sc. 30	13	30	2	161

The algorithms are launched for various fuzzy measures that correspond to typical cases of aggregation functions. The μ_{min} measure models an intolerant expert (i.e., complementary criteria), μ_{max} models a tolerant expert and μ_{mean} models the case where criteria are independents. Although μ_{min}, μ_{max} and μ_{mean} are respectively close to the min, max and the mean functions, they are not exactly equal to these functions.

5.3. Results

To evaluate the effect of propagation we use a simple propagation from the scores u_1, u_2, u_3 to the overall evaluation y (basically, only (AB1) and (AB2) are propagated). This strategy is denoted B&B. It is compared to a solving where arc-B-consistency is enforced at each node of the search tree (denoted A-B-C). In order to allow objective comparisons, we use a static chronological labeling strategy. The variables that model the date of the exams are ordered according to the number of disjunctions between exams and the number of enrolments for each exam. The algorithms are compared according to the number of backtracks and the completion time needed to optimally solve each problem.

	μ_{min}				μ_{mean}				μ_{max}			
	B&B		A-B-C		B&B		A-B-C		B&B		A-B-C	
	Btk	s	Btk	s	Btk	s	Btk	s	Btk	s	Btk	s
Sc. 12	1083	0.08	308	0.04	732	0.06	215	0.02	392	0.04	85	0.02
Sc. 15	22271	1.88	3401	0.47	237783	21.87	93289	9.13	67710	5.53	12228	1.45
Sc. 20	738013	72.65	49832	9.45	1357203	135.57	182540	33.45	349852	35.54	35011	4.77
Sc. 30	35.10^6	1:02:13	$7.7.10^6$	0:22:55	190.10^6	6:08:07	53.10^6	1:45:25	-	-	-	-

Time results are indicated either in seconds or in the h:mn:s format. The "-" symbol means that the problem couldn't be solved within 48 hours.

These results clearly show that, for any kind of fuzzy measure, propagating the Choquet constraint considerably reduces the size of the tree that is constructed by the search. Although the time effort is $O(n^2)$ for the Choquet constraint with respect to $O(n)$ for the straightforward calculation, the solving time is also considerably reduced for all measures and instances.

In addition, when the min, mean and max functions are modeled with the Choquet integral, the time used for the propagation of Choquet can be compared to the time taken by the min, sum and max constraints. Times obtained with the solver constraints (S-C) are compared with the others in the following table:

	min			mean			max		
	B&B	A-B-C	S-C	B&B	A-B-C	S-C	B&B	A-B-C	S-C
Sc. 12	0.03	0.01	0.01	0.06	0.03	0.04	0	0	0
Sc. 15	1.4	0.234	0.219	2.37	1.03	0.89	0	0	0
Sc. 20	63.5	5.48	5.17	85.7	21.44	19.45	0	0	0
Sc. 30	0:37:53	0:09:31	0:09:16	1:36:24	0:46:31	0:44:52	-	26:28:42	25:24:41

As previously observed, A-B-C is clearly better than B&B. Comparing A-B-C and S-C, we can see that the difference between the "generic" Choquet algorithms and the dedicated algorithms of each constraint remains quite small. Hence, the expressivity gained with Choquet does not generate a great loss in efficiency.

In addition, in this table we can see that in the max case, an optimal solution is immediately found by the solver, except for scenario 30 which takes very long (B&B failed to solve the problem within 2 days). In this case, the heuristics directly finds a solution that is equal to 1 on criterion "duration" (and thus is optimum for max). As no such solution exists in Sc. 30, a very large part of the search tree is explored before a solution that completely satisfies the "spreading" criterion is found. This illustrates well another difficult problem, which is the definition of search heuristics when the objective depends on several criteria.

6. Conclusion

In this paper we introduced a new approach for the solving of multi-criteria optimization problems in CP. As the elicitation of multi-criteria preferences is a very hard problem on its own, we proposed to integrate a very general model from MCDM and to benefit from the well proven methodology that has already been developed for the elicitation of its parameters.

To achieve this integration we proposed a general scheme based on the definition of a multi-criteria aggregation constraint that establishes the performance of a solution. According to general properties of multi-criteria aggregation functions, we defined some arc-B-consistency conditions with respect to constraints that model these functions. We showed that if the constraint is arc-B-consistent, then it is necessarily arc-consistent and that an algorithm that propagates once each of the consistency conditions is idempotent. These principles were applied to the case of the Choquet integral, which is a very flexible aggregation function that offers good properties for modeling preferences. We defined the Choquet global constraint and proposed propagation algorithms for modeling this function in a CP solver. This constraint has been evaluated on the multi-criteria timetabling problem. The results obtained on the optimal resolution of small instances for different kind of preference models clearly show the interest of propagation.

A first observation is that the aggregation of criteria finally enables reducing the problem to a mono-criterion optimization problem. However, the problem keeps an underlying multi-criteria structure that, in particular, considerably increases the complexity of defining good search heuristics. Some work has been led on this issue and deserves further investigations (Le Huédé et al., 2003).

References

Apt, K.R. (1998). "A Proof Theoretic View of Constraint Programming." *Fundamenta Informaticae* 34(3), 295–321.

Apt, K.R. (1999). "The Essence of Constraint Propagation." *Theoretical Computer Science* 221(1–2), 179–210.

Bana e Costa, C.A. and J.C. Vansnick. (1994). "A Theoretical Framework for Measuring Attractiveness by a Categorical Based Evaluation TecHnique (MACBETH)." In *Proc. XIth Int. Conf. on MCDM*, Coimbra, Portugal, pp. 15–24.

Barichard, V. and J.-K. Hao. (2003). "A Population and Interval Constraint Propagation Algorithm." In *LNCS 2632*, Springer, pp. 88–101.

Bistarelli, S., U. Montanari, F. Rossi, T. Schiex, G. Verfaillie, and H. Fargier. (1999). "Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison." *CONSTRAINTS* 4, 199–240.

Boutilier, C., R. Brafman, C. Domshlak, H.H. Hoos, and D. Poole. (2004). "CP-nets: A Tool for Representing and Reasoning with Conditional Ceteris Paribus Preference Statements." *Journal of Artificial Intelligence Research* 21, 135–191.

Boutilier, C., R. Brafman, C. Domshlak, H.H. Hoos, and D. Poole. (2004). "Preference-Based Constrained Optimization with CP-nets." *Computational Intelligence* 20(2), 137–157.

Choquet, G. (1953). "Theory of Capacities." *Annales de l'Institut Fourier* 5, 131–295.

Gavanelli, M. (2002). "An Algorithm for Multi-Criteria Optimization in CSPs." In Frank van Harmelen, editor, *Proceedings of ECAI-2002*, Lyon, France, IOS Press, pp. 136–140.

Grabisch, M. (1996). "The Application of Fuzzy Integrals in Multicriteria Decision Making." *European J. of Operational Research* 89, 445–456.

Grabisch, M. (2000). "The Interaction and Möbius Representations of Fuzzy Measures on Finite Spaces, k-Additive Measures: A Survey." In M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy Measures and Integrals—Theory and Applications*, Physica Verlag, pp. 70–93.

Grabisch, M. and M. Roubens. (2000). "Application of the Choquet Integral in Multicriteria Decision Making." In M. Grabisch, T. Murofushi, and M. Sugeno, editors, *Fuzzy Measures and Integrals—Theory and Applications*, Physica Verlag, pp. 348–374.

Ben Jaâfar, I., N. Khayati, and K. Ghédira. (2004). "Multicriteria Optimization in CSPs: Foundations and Distributed Solving Approach." In Christoph Bussler and Dieter Fensel, editors, *AIMSA*, volume 3192 of *LNCS*, Springer, pp. 459–468.

Junker, U. (2004). "Preference-Based Search and Multi-Criteria Optimization." *Annals of OR* 130, 75–115.

Kaymak, U. and J.M. Sousa. (2003). "Weighted Constraint Aggregation in Fuzzy Optimization." *Constraints* 8(1), 61–78, Jan.

Keeney, R.L. and H. Raiffa. (1976). *Decision with Multiple Objectives*. Wiley, New York.

Labreuche, C. and M. Grabisch. (2003). "The Choquet Integral for the aggregation of Interval Scales in Multicriteria Decision making." *Fuzzy Sets and Systems* 137(1), 11–26.

Labreuche, C. and F. Le Huédé. (2005). "MYRIAD: a Tool Suite for MCDA." In *EUSFLAT'05*, Barcelona, Spain.

Le Huédé, F. (2003). *Intégration d'un modèle d'Aide à la Décision Multicritère en Programmation Par Contraintes*. PhD thesis, Universitée Paris 6.

Le Huédé, F., M. Grabisch, C. Labreuche, and P. Savéant. (2003). "Multicriteria Search in Constraint Programming." In *Proceedings of CP-AI-OR'03*, Montréal, Canada, pp. 291–304.

Lhomme, O. (1993). "Consistency Techniques for Numerical CSPs." In *Proceedings of IJCAI 1993*, Chambery, France, pp. 232–238.

Marichal, J.L. (1998). *Aggregation Operators for Multicriteria Decision Aid*. PhD thesis, University of Liège.

Michel, C., M. Rueher, and Y. Lebbah. (2001). "Solving Constraints Over Floationg-Point Numbers." In *Proceedings of CP-2001*, Springer.

Museux, N., L. Jeannin, P. Savéant, F. Le Huédé, F.-X. Josset, and J. Mattioli. (2003). "Claire/Eclair: Un environnement de modélisation et de résolution pour des applications d'optimisation combinatoires embarquées." In *Proceedings of JFPLC'03*, Amiens, France.

Prestwich, S., F. Rossi, K.B. Venable, and T. Walsh. (2005). "Constraint-Based Preferential Optimization." In *AAAI*, pp. 461–466.

Pearl Pu and Boi Faltings. (2004). "Decision Tradeoff using Example-Critiquing and Constraint Programming." *Constraints* 9(4), 289–310.

Refalo, P. (1999). "Tight Cooperation and its Application in Piecewise Linear Optimisation." In *Proceedings of CP-99*, Springer, pp. 375–389.

Sugeno, M. (1974). *Theory of fuzzy integrals and its applications*, PhD thesis, Tokyo Institute of Technology.

Zhang, Y. and R. H.C. Yap. (2000). "Arc Consistency on n-ary Monotonic and Linear Constraints." In *Proceedings of CP-2000*, Singapore, pp. 470–483.