GAI Networks for Decision Making under Certainty

Christophe Gonzales and **Patrice Perny**
LIP6 – Paris 6 University – France
{Christophe.Gonzales,Patrice.Perny}@lip6.fr

Abstract

This paper deals with preference elicitation and preference-based optimization in the context of multiattribute utility theory under certainty. We focus on the generalized additive decomposable utility model which allows interactions between attributes while preserving some decomposability. We first present a systematic elicitation procedure for such utility functions. This procedure relies on a graphical model called a *GAI-network* which is used to represent and manage independences between attributes, just as junction graphs model independences between random variables in Bayesian networks. Then, we propose an optimization procedure relying on this network to compute efficiently the solution of optimization problems over a product set.

1 Introduction

The development of decision support systems has stressed the need for preference models to handle user's preferences and perform preference-based recommendation tasks.

In this respect, current works in preference modeling and decision theory aim at proposing compact preference models achieving a good compromise between two conflicting aspects: on the one hand, the need for sufficiently flexible models to describe sophisticated decision behaviors, on the other hand, the practical necessity of keeping the elicitation effort at an admissible level as well as the need for efficient procedures to solve preference-based optimization problems. As an example, let us mention the recent advances on qualitative preference models such as CP-nets [Boutilier *et al.*, 2004a; 2004b]. Such models are naturally suited to simple applications (e.g. recommender systems to buy books on the web) in which preferences can easily be approximated by lexicographic rules on attributes with small domains. However, in more involved decision problems (e.g. modeling the fine knowledge of an expert in order to automatize an important decision task) utilities might significantly outperform qualitative models due to their higher descriptive power [Boutilier *et al.*, 2001].

In the literature, different quantitative models based on utilities have been developped to take into account different preference structures. The most widely used model assumes a special kind of independence among attributes called "mutual preferential independence" which ensures that the preferences are representable by an additively decomposable utility function [Krantz *et al.*, 1971; Bacchus and Grove, 1995]. Such decomposability makes the elicitation process easy to perform as we shall see in Section 3. However, in practice, preferential independence may fail to hold as it rules out any interaction among attributes. Generalizations of preferential independence have thus been investigated that extend the range of application of utilities. For instance *utility independence* on every attribute leads to a more sophisticated form of utility called *multilinear utility* [Bacchus and Grove, 1995]. Multilinear utilities are more general than additive utilities but many interactions between attributes still cannot be taken into account by such functionals. To increase the descriptive power of such models, GAI (generalized additive independence) decompositions introduced by [Fishburn, 1970] allow more general interactions between attributes [Bacchus and Grove, 1995] while preserving some decomposability. Such a decomposition has been used to endow CP-nets with utility functions (UCP-nets) both in uncertainty [Boutilier *et al.*, 2001] and in certainty [Brafman *et al.*, 2004].

In the same direction [Gonzales and Perny, 2004] proposes a general procedure to assess a GAI decomposable utility function in the context of decision making under risk. The elicitation procedure is directed by the structure of a new graphical model called a GAI network. The procedure consists of a sequence of questions involving simple lotteries to capture the basic features of the DM attitude under risk. As such, it does not fit for eliciting utilities under certainty. The aim of this paper is to complete this study by investigating the potential contribution of GAI networks in the context of decision making under certainty. Section 2 recalls the basics of GAI networks. The elicitation under certainty is then explained in Section 3 and an efficient process for solving optimization queries is introduced in Section 4.

2 GAI networks

Before describing GAI networks, we shall introduce some notations and assumptions. Throughout the paper, $\succsim$ denotes a decision maker's (DM) preference relation, which we assume to be a weak order, over some set $\mathcal{X}$. $x \succsim y$ means that for the DM x is at least as good as y. $\succ$ refers to the asymmetric part of $\succsim$ and $\sim$ to the symmetric one. In practical situations, $\mathcal{X}$ is often described by a set of attributes. For simplicity, we assume that $\mathcal{X}$ is the Cartesian product of the domains of these attributes, although extensions to general subsets are possible [Chateauneuf and Wakker, 1993]. In the rest of the paper, uppercase letters (possibly subscripted) such as A, B, X_1 denote both attributes and their domains (as this is unambiguous and it simplifies the notation). Unless otherwise mentioned, (possibly superscripted) lowercase letters denote values of the attribute with the same uppercase letters: x, x^1 (resp. x_i, x_i^1) are thus values of X (resp. X_i).

Under mild hypotheses [Debreu, 1964], it can be shown that $\succsim$ is representable by a utility, i.e., there exists a function $u : \mathcal{X} \mapsto \mathbb{R}$ such that $x \succsim y \Leftrightarrow u(x) \geq u(y)$ for all $x, y \in \mathcal{X}$. As preferences are specific to each individual, utility functions must be elicited for each DM, which is usually impossible due to the combinatorial nature of $\mathcal{X}$. Fortunately, DM's preferences usually have an underlying structure induced by independences among attributes that substantially decreases the elicitation burden. In this paper, we focus on a particular decomposition of utilities defined as follows:

Definition 1 (GAI decomposition) *Let $\mathcal{X} = \prod_{i=1}^n X_i$. Let $Z_1, \ldots, Z_k$ be some subsets of $N = \{1, \ldots, n\}$ such that $N = \cup_{i=1}^k Z_i$. For every i, let $X_{Z_i} = \prod_{j \in Z_i} X_j$. Utility $u(\cdot)$ representing $\succsim$ is GAI-decomposable w.r.t. the X_{Z_i}'s iff there exist functions $u_i : X_{Z_i} \mapsto \mathbb{R}$ such that:*

$$u(x) = \textstyle\sum_{i=1}^k u_i(x_{Z_i}), \text{ for all } x = (x_1, \ldots, x_n) \in \mathcal{X},$$

where x_{Z_i} denotes the tuple constituted by the x_j's, $j \in Z_i$.

GAI decompositions can be represented by graphical structures we call *GAI networks* which are essentially similar to the junction graphs used for Bayesian networks [Jensen, 1996; Cowell *et al.*, 1999]:

Definition 2 (GAI network) *Let $\mathcal{X} = \prod_{i=1}^n X_i$. Let $Z_1, \ldots, Z_k$ be some subsets of $N = \{1, \ldots, n\}$ such that $\bigcup_{i=1}^k Z_i = N$. Assume that $\succsim$ is representable by a GAI-decomposable utility $u(x) = \sum_{i=1}^k u_i(x_{Z_i})$ for all $x \in \mathcal{X}$. Then a GAI network representing $u(\cdot)$ is an undirected network $G = (V, E)$, satisfying the following properties:*

1. *$V = \{X_{Z_1}, \ldots, X_{Z_k}\}$;*
2. *For every $(X_{Z_i}, X_{Z_j}) \in E$, $Z_i \cap Z_j \neq \emptyset$. Moreover, for every pair of nodes X_{Z_i}, X_{Z_j} such that $Z_i \cap Z_j = T_{ij} \neq \emptyset$, there exists a path in G linking X_{Z_i} and X_{Z_j} such that all of its nodes contain all the indexes of T_{ij} (Running intersection property).*

Nodes of V are called cliques. *Every edge $(X_{Z_i}, X_{Z_j}) \in E$ is labeled by $X_{T_{ij}} = X_{Z_i \cap Z_j}$ and is called a* separator.

Throughout this paper, cliques will be drawn as ellipses and separators as rectangles. Moreover, we shall only be interested in GAI trees, i.e., in singly-connected GAI networks. As mentioned in [Gonzales and Perny, 2004], this is not restrictive as general GAI networks can always be compiled into GAI trees. For any GAI decomposition, by Definition 2 the cliques of the GAI network should be the sets of variables of the subutilities. For instance, if $u(a, b, c, d, e, f, g) = u_1(a, b) + u_2(c, e) + u_3(b, c, d) + u_4(b, d, f) + u_5(b, g)$ then, as shown in Figure 1, the cliques are: AB, CE, BCD, BDF, BG. By property 2 of Definition 2 the set of edges of a GAI network can be determined by any algorithm preserving the running intersection property (see the Bayesian network literature on this matter [Cowell *et al.*, 1999]).

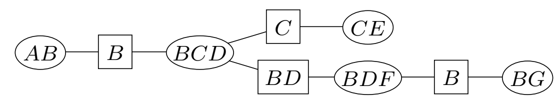

Figure 1: A GAI tree

3 Elicitation

Elicitation of GAI-decomposable utilities is closely related to that of additive utilities. Hence the beginning of this section is devoted to the latter. Then the process is extended to GAI.

3.1 Additive Utilities

Unlike decision under uncertainty or under risk, utilities under certainty are not necessarily unique up to strictly positive affine transforms [Adams, 1965; Gonzales, 2003]. However, such uniqueness property usually greatly simplifies the elicitation process. Hence, the property below, which ensures this uniqueness, will be assumed throughout this section. Note however, it can be easily dispensed with [Gonzales, 2003].

Definition 3 (restricted solvability) *Let $\mathcal{X} = \prod_{i=1}^n X_i$. For every i, if $(x_1^0, \ldots, x_{i-1}^0, x_i^0, x_{i+1}^0, \ldots, x_n^0) \precsim (x_1, \ldots, x_n) \precsim (x_1^0, \ldots, x_{i-1}^0, x_i^1, x_{i+1}^0, \ldots, x_n^0)$, then there exists $x_i^2 \in X_i$ such that $(x_1^0, \ldots, x_{i-1}^0, x_i^2, x_{i+1}^0, \ldots, x_n^0) \sim (x_1, \ldots, x_n)$.*

This property usually implies that $\mathcal{X}$ contains numerous elements, if not an infinite number. However, this restricts neither the potential field of applications nor the applicability of the elicitation process. Indeed, if an attribute is not solvable, then its domain can always be enriched so that solvability holds as long as only the projection on the original space is used when making decisions. As for the applicability of elicitation, $\mathcal{X}$ being infinite is not a problem because indifference curves —the sets of elements in a same indifference class— are usually very smooth and thus can be easily approximated knowing only a limited number of points.

Under certainty, the central idea for elicitation lies in the construction of standard sequences: let $\succsim$ be a preference relation on $X_1 \times X_2$ representable by an additive utility $u_1(\cdot) + u_2(\cdot)$. Let $x_1^0 \in X_1$, $x_2^0, x_2^1 \in X_2$ be arbitrary points. As by solvability utilities are unique up to strictly positive affine transforms, we may assume without loss of generality that $(x_1^0, x_2^1) \succ (x_1^0, x_2^0)$, $u_1(x_1^0) = u_2(x_2^0) = 0$ and $u_2(x_2^1) = 1$. Let $x_1^1 \in X_1$ be such that $(x_1^1, x_2^0) \sim (x_1^0, x_2^1)$ (see Figure 2 which represents indifference curves in space $X_1 \times X_2$), then $u_1(x_1^1) = 1$. More generally, for any i, let x_1^i, x_1^{i+1} be such that $(x_1^{i+1}, x_2^0) \sim (x_1^i, x_2^1)$. Then $u_1(x_1^i) = i$. Sequence (x_1^i) is called a standard sequence:

Definition 4 (standard sequence) *For any set N of consecutive integers, $\{x_1^k, k \in N\}$ is a standard sequence w.r.t. X_1 iff there exist $x^0 = (x_2^0, \ldots, x_n^0)$ and $x^1 = (x_2^1, \ldots, x_n^1)$ such that* Not$[(x_1^0, x_2^0, \ldots, x_n^0) \sim (x_1^0, x_2^1, \ldots, x_n^1)]$ *and for all $k, k+1 \in N$, $(x_1^{k+1}, x_2^0, \ldots, x_n^0) \sim (x_1^k, x_2^1, \ldots, x_n^1)$.*

$\{x^0, x^1\}$ is called the mesh of the standard sequence. Similar definitions hold for the other X_i's, $i > 1$.

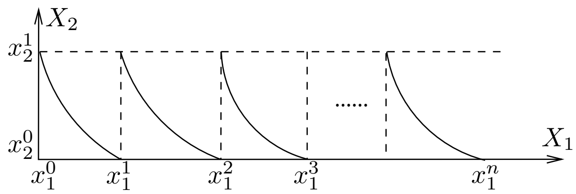

Figure 2: A standard sequence w.r.t. X_1

Thus, eliciting $u_1(\cdot)$ merely amounts to choosing two values for the other attributes, say x_2^0, x_2^1, assigning $u_1(x_1^0) = u_2(x_2^0) = 0$ and $u_2(x_2^1) = 1$, and constructing a standard sequence of mesh $\{x_2^0, x_2^1\}$. This results in an additive utility defined on $X_1 \times \{x_2^0, x_2^1\}$. Similarly an additive utility $v(\cdot) = v_1(\cdot) + v_2(\cdot)$ can be defined on $\{x_1, x_1'\} \times X_2$ by selecting two values, $x_1, x_1' \in X_1$, assigning $v_2(x_2^0) = v_1(x_1) = 0$ and $v_1(x_1') = 1$, and constructing a standard sequence of mesh $\{x_1, x_1'\}$. Constructing separately $u_1(\cdot)$ and $v_2(\cdot)$ does not ensure that $u_1(\cdot) + v_2(\cdot)$ is a utility on $X_1 \times X_2$. Indeed, if $w(\cdot) = w_1(\cdot) + w_2(\cdot)$ is known to be an additive utility on $X_1 \times X_2$ then, by uniqueness up to strictly positive affine transforms, there exist $\alpha > 0$, $\beta > 0$ and $\gamma, \delta \in \mathbb{R}$ such that the restrictions of $w_1(\cdot) + w_2(\cdot)$ on $X_1 \times \{x_2^0, x_2^1\}$ and $\{x_1, x_1'\} \times X_2$ are equal to $\alpha[u_1(\cdot) + u_2(\cdot)] + \gamma$ and $\beta[v_1(\cdot) + v_2(\cdot)] + \delta$ respectively. This implies in particular that $\alpha u(\cdot) + \gamma$ and $\beta v(\cdot) + \delta$ must match on $\{x_1, x_1'\} \times \{x_2^0, x_2^1\}$, hence resulting in constraints on the admissible values of $\alpha, \beta, \gamma, \delta$. As multiplying $w(\cdot)$ by $1/\alpha$ and subtracting γ does not alter its representability, $u_1(\cdot) + u_2(\cdot)$ and $(\beta/\alpha)[v_1(\cdot) + v_2(\cdot)] + \delta - \gamma$ must match on $\{x_1, x_1'\} \times \{x_2^0, x_2^1\}$. The value of the ratio β/α follows directly from:

$$u_1(x_1') + u_2(x_2^0) = (\beta/\alpha)[v_1(x_1') + v_2(x_2^0)] + \delta - \gamma,$$
$$u_1(x_1) + u_2(x_2^0) = (\beta/\alpha)[v_1(x_1) + v_2(x_2^0)] + \delta - \gamma,$$

and is equal to $[u_1(x_1') - u_1(x_1)]/[v_1(x_1') - v_1(x_1)]$. Constants γ and δ need not be determined as adding constants to a utility does not alter its representability. Hence $u_1(\cdot) + (\beta/\alpha)v_2(\cdot)$ is guaranteed to be an additive utility representing $\succsim$ on $X_1 \times X_2$ and the elicitation is completed. Note that, without the above uniqueness, an overall utility cannot be constructed at a low cost from marginal utilities. It would indeed require asking an exponential number of questions involving tuples differing by many if not all their attributes.

Of course, the above construction process can be generalized to spaces with more than 2 attributes: first, for every attribute X_i, a subutility $u_i(\cdot)$ is elicited using a standard sequence, then $u_2(\cdot)$ is rescaled to fit with $u_1(\cdot)$, $u_3(\cdot)$ is rescaled to fit with $u_1(\cdot) + u_2(\cdot)$, and so on.

3.2 Generalized Additive Utilities

Eliciting a GAI-decomposable utility is similar in essence to eliciting an additive utility: first subutilities on cliques are elicited using standard sequences, then they are rescaled. In the preceding section, such a process worked finely because additive utilities are unique up to strictly positive affine transforms. Unfortunately, even under restricted solvability such uniqueness property does not apply to GAI-decomposable utilities. But this is easily fixed by the following proposition:

Proposition 1 *Let $\mathcal{X} = \prod_i X_i$ and let $u(\cdot)$ be a utility decomposable according to a GAI tree $\mathcal{G} = (\mathcal{C}, \mathcal{E})$ where $\mathcal{C} = \{X_{C_1}, \ldots, X_{C_k}\}$, i.e., $u(x) = \sum_{i=1}^{|\mathcal{C}|} u_i(x_{C_i})$. Assume that cliques C_i's are ordered from the outer cliques of $\mathcal{G}$ to the inner ones, i.e., they are such that for all i, if $\mathcal{G}_{\mathcal{C}_i}$ denotes the subgraph of $\mathcal{G}$ induced by $\mathcal{C}_i = \mathcal{C} \backslash \{C_j : j < i\}$, then $\mathcal{G}_{\mathcal{C}_i}$ is connected. Let x^0 be an arbitrary element of $\mathcal{X}$. Then:*

1. *For every $i \in \{1, \ldots, |\mathcal{C}| - 1\}$, clique X_{C_i} has exactly one neighbor, denoted by $X_{C_{n(i)}}$, in $\mathcal{G}_{\mathcal{C}_i}$.*
2. *Let $S_i = C_i \cap C_{n(i)}$ and $D_i = C_i \backslash S_i$. There exists a utility v GAI-decomposable according to $\mathcal{G}$ such that for all $i < |\mathcal{C}|$, and all x_{S_i}, $v_i(x_{D_i}^0, x_{S_i}) = 0$ and such that $v_{|\mathcal{C}|}(x_{C_{|\mathcal{C}|}}^0) = 0$. Moreover such GAI-decomposable utility is unique up to strictly positive linear transforms.*

Let us illustrate this proposition on the GAI network of Figure 1: $\mathcal{C}$ can be set to $\{AB, CE, BCD, BDF, BG\}$ because AB and CE are outer cliques as their removal keeps the graph connected. Similarly, after removing AB and CE, BCD becomes an outer clique as its additional removal keeps the graph connected, and so on. Property 1 states that clique AB is connected to only one adjacent clique, here BCD; after removing both AB and CE, BCD is connected to only BDF, etc. Property 2 states that if $(a^0, b^0, c^0, d^0, e^0, f^0, g^0)$ is an arbitrary element of $\mathcal{X}$, then there exists a GAI-decomposable utility $v(a, b, c, d, e, f, g) = v_1(a, b) + v_2(c, e) + v_3(b, c, d) + v_4(b, d, f) + v_5(b, g)$ such that $v_1(a^0, b) = v_2(c, e^0) = v_3(b, c^0, d) = v_4(b, d^0, f^0) = v_5(b^0, g^0) = 0$. The idea behind this property is simple: the X_{S_i}'s are attributes belonging to separators, thus the property can be established by transferring via separator X_{S_i} some quantity depending only on the X_{S_i}'s from one clique to its neighbor. For instance, assume that $v_1(a^0, b) \neq 0$ for all b, then substituting $v_1(a, b)$ by $v_1(a, b) - v_1(a^0, b)$ for all $(a, b) \in A \times B$ as well as $v_3(b, c, d)$ by $v_3(b, c, d) + v_1(a^0, b)$ for all $(b, c, d) \in B \times C \times D$ yields the property for $v_1(\cdot)$. The same can then be applied on the second clique of $\mathcal{C}$ and by induction to all the cliques in $\mathcal{C}$. As for the last clique, subtracting constant $v_5(b^0, g^0)$ to $v_5(b, g)$ and property 2 holds.

Now we can show how the elicitation can be conducted on the GAI network of Figure 1. Let $(a^0, b^0, c^0, d^0, e^0, f^0, g^0)$ be an arbitrary element of the preference space. By Proposition 1, there exists a GAI-decomposable utility such that:

$$v_1(a^0, b) = 0 \qquad v_2(c, e^0) = 0$$
$$v_3(b, c^0, d) = 0 \qquad v_4(b, d^0, f^0) = 0 \qquad v_5(b^0, g^0) = 0.$$

The elicitation process consists in constructing $v_i(\cdot)$'s on each X_{C_i} in the order where they are appear in $\mathcal{C}$, i.e., here, AB, CE, BCD, BDF, BG. Then each newly constructed subutility is rescaled to fit with the previously constructed ones.

First construct $v_1(\cdot)$ on $A \times B$. Let b^1 be any value in B. Note that the restriction of $v(\cdot)$ on hyperplane $B = b^1$ is an additive utility: indeed $v_1(a, b^1)$ and $[v_2(c, e) + v_3(b^1, c, d) + v_4(b^1, d, f) + v_5(b^1, g)]$ are functions of A and $C \times D \times E \times F \times G$ respectively. Consequently eliciting

$v_1(\cdot, b^1)$ just requires constructing a standard sequence w.r.t. A. Let $\{(b^1, c^1, d^1, e^1, f^1, g^1), (b^1, c^2, d^2, e^2, f^2, g^2)\}$ be the mesh of this sequence. As we do not know the values of the $\sum_{i>2} v_i(\cdot)$'s for the elements of the mesh, we will assume they are 0 and 1 respectively, hence the utility we will construct is not $v(\cdot)$ but rather $w(\cdot)$ such that $\alpha w(\cdot) + \gamma = v(\cdot)$ for some constants α and γ. So assign:

$$w_1(a^0, b^1) = 0,$$
$$w_2(c^1, e^1) + w_3(b^1, c^1, d^1) + w_4(b^1, d^1, f^1) + w_5(b^1, g^1) = 0,$$
$$w_2(c^2, e^2) + w_3(b^1, c^2, d^2) + w_4(b^1, d^2, f^2) + w_5(b^1, g^2) = 1.$$

Then using the standard sequence, $w_1(a, b^1)$ can be estimated for any $a \in A$ as described previously. Similarly, for another value b^2 of B, a mesh $\{(b^2, c^3, d^3, e^3, f^3, g^3), (b^2, c^4, d^4, e^4, f^4, g^4)\}$ can be chosen and a utility $w'(\cdot)$ such that $\beta w'(\cdot) + \delta = v(\cdot)$ for some β and δ and such that:

$$w'_1(a^0, b^2) = 0,$$
$$w'_2(c^3, e^3) + w'_3(b^2, c^3, d^3) + w'_4(b^2, d^3, f^3) + w'_5(b^3, g^3) = 0,$$
$$w'_2(c^4, e^4) + w'_3(b^2, c^4, d^4) + w'_4(b^2, d^4, f^4) + w'_5(b^4, g^4) = 1.$$

can be elicited. Now there remains to rescale $w'_1(\cdot)$ so as to fit with $w_1(\cdot)$. In other words, ratio β/α need be evaluated. This can be simply achieved by asking the DM for given values $a, a'' \in A$ to exhibit values a' and a''' of A such that:

$$(a, b^1, c^1, d^1, e^1, f^1, g^1) \sim (a''', b^2, c^3, d^3, e^3, f^3, g^3),$$
$$(a', b^1, c^1, d^1, e^1, f^1, g^1) \sim (a'', b^2, c^3, d^3, e^3, f^3, g^3).$$

Then $\beta/\alpha = [w_1(a', b^1) - w_1(a, b^1)]/[w'_1(a'', b^2) - w'_1(a''', b^2)]$. Thus, function $v_1(a, b)$ defined by:

$$v_1(a, b) = \begin{cases} w_1(a, b) & \text{if } b = b^1 \\ (\beta/\alpha) w'_1(a, b) & \text{if } b = b^2 \end{cases}$$

is a utility function over $A \times \{b^1, b^2\}$. The same process can be used to extend the definition of $v_1(\cdot)$ over $A \times B$. Here again, it should be mentioned that only a few values of B are needed as $v_1(\cdot)$ can be approximated using the smoothness of the indifference curves.

Once $v_1(\cdot)$ has been elicited, a similar process can be used for eliciting a utility $w_2(\cdot)$ over $C \times E$. Here again the key point is to use the additive decomposition given fixed values of separator C. And by asking questions to the DM involving only tuples on hyperplane $A = a^0$, function $v_1(\cdot)$ can be removed from the equations as $v_1(a^0, b) = 0$ for all b's. In graphical terms, this corresponds to removing clique AB from the GAI tree and performing a new elicitation process on $B \times C \times D \times E \times F \times G$. Of course, $v_1(\cdot) + w_2(\cdot)$ is not guaranteed to be a utility function as the scale of $v_1(\cdot)$ may not fit that of $w_2(\cdot)$. Hence $w_2(\cdot)$ need be rescaled. This can be achieved by asking one question involving only the attributes that do not belong to separators (the X_{S_i}'s of Proposition 1), i.e., A and E: let $a \in A$ and $e \in E$ be such that:

$$(a, b^0, c^0, d^0, e^0, f^0, g^0) \sim (a^0, b^0, c^0, d^0, e, f^0, g^0).$$

If $v_2(\cdot) = (v_1(a, b^0)/w_2(c^0, e)) w_2(\cdot)$, then $v_1(\cdot) + v_2(\cdot)$ is a GAI-decomposable utility.

Once $v_1(\cdot)$ and $v_2(\cdot)$ have been elicited, clique CE can also be removed from the GAI tree, hence resulting in Figure 3. A utility $w_3(\cdot)$ can be elicited in this graph asking questions involving only tuples on hyperplane $A = a^0$ and $E = e^0$. Here separator X_{S_i} of Proposition 1 is BD, so $w_3(\cdot)$ should first be constructed on hyperplanes where B and D's values are fixed, i.e., $w_3(b^i, \cdot, d^i) : C \mapsto \mathbb{R}$ should be constructed for several distinct values (b^i, d^i). Then they are rescaled to form a global utility $w_3(\cdot)$ on $B \times C \times D$. Finally, this utility need also be rescaled to fit with $v_1(\cdot)$ and $v_2(\cdot)$. As for CE it can be achieved asking one question. This one can involve distinct values of either C and A (thus fitting $w_3(\cdot)$ with $v_1(\cdot)$) or C and E (thus fitting $w_3(\cdot)$ with $v_2(\cdot)$). This choice makes no difference on a computational point of view. Assume that we chose the (C, A) pair. Then:

$$(a', b^0, c^0, d^0, e^0, f^0, g^0) \sim (a^0, b^0, c', d^0, e^0, f^0, g^0)$$

implies that $v_1(\cdot) + v_2(\cdot) + (v_1(a', b^0)/w_3(b^0, c', d^0)) w_3(\cdot)$ is a GAI utility. The process can be applied until all cliques are eliminated from the graph, hence resulting in a GAI-decomposable utility over the whole $\mathcal{X}$.

Figure 3: The GAI tree after removing cliques AB and CE

Finally, to conclude this section, we should mention that this process can be easily extended to utilities decomposable according to GAI forests, i.e., to sets of GAI trees (such as in Figure 4). Indeed, the sum of the utilities on each connected component forms an additive utility on $\mathcal{X}$. Hence, eliciting a utility on this space merely amounts to applying the above process on each connected component and, then, rescaling all these subutilities as we would do for a usual additive utility.

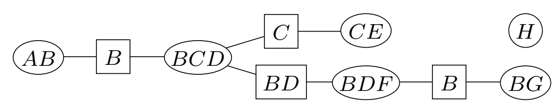

Figure 4: A GAI forest

4 GAI-based optimization

Once a utility function of the DM has been elicited, it can be used for recommendation tasks. Several type of queries might be of interest, for instance:

- *overall optimization queries:* find the preferred tuple over the entire product set $\mathcal{X}$.
- *restricted optimization queries:* find the preferred tuple over the entire product set $\mathcal{X}$ conditionally to the values of a given subset of attributes.
- *preference queries:* find which among a given pair of tuples $(x, y) \in \mathcal{X} \times \mathcal{X}$ is preferred by the DM.

We now focus on queries of the first type (overall optimization) and present an efficient procedure to solve them. Restricted optimization being a particular specification of this problem, our procedure can be directly adapted to solve it as well. The third type of queries is not critical on a computational viewpoint as a preference query for a given pair (x, y) can be solved at a low cost by simply computing and comparing $u(x)$ and $u(y)$. However, extending such a pairwise

$u_1(a,b)$	b^0	b^1
a^0	8	2
a^1	4	3
a^2	1	7

$u_2(c,e)$	e^0	e^1	e^2
c^0	6	3	5
c^1	3	4	0

$u_3(b^0,c,d)$	d^0	d^1
c^0	0	2
c^1	5	1

$u_3(b^1,c,d)$	d^0	d^1
c^0	7	1
c^1	2	4

$u_4(b^0,d,f)$	f^0	f^1
d^0	4	2
d^1	3	8

$u_4(b^1,d,f)$	f^0	f^1
d^0	5	8
d^1	9	0

$u_5(b,g)$	g^0	g^1
b^0	0	9
b^1	6	4

$u_6(h)$	h^0	h^1	h^2	h^3	h^4
	6	3	4	1	10

Figure 5: Utility tables for u

comparison to all pairs $(x,y) \in \mathcal{X} \times \mathcal{X}$ is prohibitive in the finite case and is simply not feasible in the infinite case. Hence another approach is needed to perform overall optimization.

Fortunately, the GAI decomposability allows the computational cost of the overall optimization task to be kept at a very admissible level. The idea is to take advantage of the structure of the GAI network to efficiently decompose the query problem into a sequence of local optimizations.

For the clarity of presentation, the overall optimization procedure is introduced on a small example with a finite Cartesian product, but it obviously generalizes to the infinite case.

Consider a decision problem where alternatives are described by 8 attributes. Assume the overall optimization is performed over the feasible set $\mathcal{X} = A \times B \times C \times D \times E \times F \times G \times H$ with $A = \{a^0, a^1, a^2\}, B = \{b^0, b^1\}, C = \{c^0, c^1\}, D = \{d^0, d^1\}, E = \{e^0, e^1, e^2\}, F = \{f^0, f^1\}, G = \{g^0, g^1\}, H = \{h^0, h^1, h^2, h^3, h^4\}$. The DM's preferences are represented by a GAI-decomposable utility defined, for any tuple (a,b,c,d,e,f,g,h) by:

$$u(a,b,c,d,e,f,g,h) = u_1(a,b) + u_2(c,e) + u_3(b,c,d) + u_4(b,d,f) + u_5(b,g) + u_6(h)$$

where the u_i's are given by Figure 5. Remark that utility u is completely characterized by only 37 integers whereas storing u in extension requires $|\mathcal{X}| = 1440$ integers. When attributes are continuous, the tables of Figure 5 are substituted by functions providing an analytical representation of u. Figure 4 depicts the GAI network representing u's decomposition.

We now introduce the optimization procedure for this case. Let $\mathcal{H}'$ represent the Cartesian product of the domains of all the attributes but H. Remark first that for any $(h', h) \in \mathcal{H}' \times H$, $u(h', h)$ can be rewritten as $v(h') + u_6(h)$. Hence:

$$\max_{x \in \mathcal{X}} u(x) = \max_{h \in \mathcal{H}'} v(h') + \max_{h \in H} u_6(h).$$

The optimization of u_6 being straightforward, let us discuss the optimization of v. Let $\mathcal{A}'$ be the Cartesian product of the domains of all the attributes involved in $\mathcal{H}'$ but A. The product structure of $\mathcal{X}$ implies that:

$$\max_{x \in \mathcal{H}'} v(x) = \max_{y \in \mathcal{A}'} \max_{a \in A} v(a,y).$$

Since $\mathcal{A}'$ is itself a product set, the same remarks apply to $\mathcal{A}'$. If $\mathcal{B}'$ represents the Cartesian product of the domains of all the attributes involved in $\mathcal{A}'$ but B, then:

$$\max_{x \in \mathcal{H}'} v(x) = \max_{b \in B} \max_{z \in \mathcal{B}'} \max_{a \in A} v(a,b,z).$$

Here $v(a,b,z)$ can be rewritten as $v_1(a,b) + v_1'(b,z)$ where v_1' is a utility on $\mathcal{A}'$. Hence, denoting $v_1^*(b) = \max_{a \in A} v_1(a,b)$ and $v_1'^*(b) = \max_{z \in \mathcal{B}'} v'(b,z)$ we get:

$$\max_{x \in \mathcal{H}'} v(x) = \max_{b \in B} \{v_1^*(b) + v_1'^*(b)\}.$$

Now, assuming that utility $v_1'^*(b)$ has been previously computed for all $b \in B$ and stored on BCD, the optimal solution can be simply obtained using the following 4 step procedure:

1. on clique AB, compute $v_1^*(b)$ for all $b \in B$ and send the result as a message to clique BCD,
2. on clique BCD, compute the optimal value of b defined by $b^* = arg\max_{b \in B}\{v_1^*(b) + v_1'^*(b)\}$
3. send message b^* to clique AB so as to determine $a^* = arg\max_{a \in A} v_1(a, b^*)$
4. on BCD, determine $z^* = arg\max_{z \in \mathcal{B}'} v_1'(b^*, z)$.

At the end of this process, (a^*, b^*, z^*) is optimal for v. Hence, setting $h^* = arg\max_{h \in H} u_6(h)$, (a^*, b^*, z^*, h^*) is optimal for u. Remember that, as a preprocessing, $v_1'^*(b)$ need be computed by optimizing $v_1'(b,z)$ over $\mathcal{B}'$ for every fixed $b \in B$. But the decomposition $v_1'(b,c,d,e,f,g) = v_2(c,e) + v_2'(b,d,e,f,g)$ suggests that this can be computed efficiently by exploiting the reduced GAI tree resulting from clique AB's deletion. This requires the optimization of $v_2'(b,d,e,f,g)$ for any fixed c that, in turn, can be performed efficiently using a decomposition such as $v_2'(b,d,e,f,g) = u_5(b,g) + u_5'(b,d,e,f)$ on the reduced GAI tree where cliques AB and BG have been deleted, and so on recursively. The details of computations in our example are given below:

Step 1: on clique AB, compute $v_1^*(b) = \max_{a \in A} v_1(a,b)$ for all $b \in B$ and send the result as a message to clique BCD,

Step 2: on clique CE, compute $v_2^*(c) = \max_{e \in E} v_2(c,e)$ for all $c \in C$ and send the result as a message to clique BCD,

Step 3: on clique BCD, aggregate messages v_1^* and v_2^* to v_3 by computing $v_3^*(b,c,d) = v_3(b,c,d) + v_1^*(b) + v_2^*(c)$ for all $(b,c,d) \in B \times C \times D$ and store the result on clique BCD,

Step 4: on clique BG, compute $v_5^*(b) = \max_{g \in G} v_5(b,g)$ for all $b \in B$ and send the result as a message to clique BDF,

Step 5: on clique BDF, aggregate message v_5^* to v_4 by computing $v_4^*(b,d,f) = v_4(b,d,f) + v_5^*(b)$ for all $(b,d,f) \in B \times D \times F$. Then compute $v_4^{**}(b,d) = \max_{f \in F} v_4^*(b,d,f)$ for all $(b,d) \in B \times D$ and send the result to clique BCD,

Step 6: on clique BCD, compute $v_3^{**}(b,d) = \max_{c \in C} v_3^*(b,c,d)$ for all $(b,d) \in B \times D$ and aggregate the result with message v_4^{**} by computing $v_3^{***}(b,d) = v_3^{**}(b,d) + v_4^{**}(b,d)$ for all $(b,d) \in B \times D$.

Step 7 : on clique H, compute $u_6^* = \max_{h \in H} u_6(h)$

The message sent are given in Figure 6.

Step 7 shows that function u_6 is optimal for h^4 with 10 points. Step 6 shows that function $v_3^{***}(b,d)$ is optimal for

$v_1^*(b)$	b^0	b^1
	8	7

$v_2^*(c)$	c^0	c^1
	6	4

$v_5^*(b)$	b^0	b^1
	9	6

$v_3^*(b^0, c, d)$	d^0	d^1
c^0	14	16
c^1	17	13

$v_3^*(b^1, c, d)$	d^0	d^1
c^0	20	14
c^1	13	15

$v_4^*(b^0, d, f)$	f^0	f^1
d^0	13	11
d^1	12	17

$v_4^*(b^1, d, f)$	f^0	f^1
d^0	11	14
d^1	15	6

$v_3^{**}(c, d)$	d^0	d^1
c^0	17	16
c^1	20	15

$v_4^{**}(b, d)$	d^0	d^1
b^0	13	17
b^1	14	15

$v_3^{***}(b, d)$	d^0	d^1
b^0	30	33
b^1	34	30

Figure 6: Messages sent in the GAI-network

(b^1, d^0) with 34 points. Hence the optimal value for u is 10 + 34 = 44 points. In order to recover the optimal solution, optimality of components (b^1, d^0) known on clique BCD must be sent to neighbors cliques and propagated in the network so as to complete the solution. More precisely, the sequence of messages is the following:

1. message (b^1, d^0) is sent from BCD to BDF. On BDF we obtain $f^1 = arg\max_{f \in F} u_5(b^1, d^0, f)$ which gives the optimal choice in F.
2. message b^1 is sent from BDF to BG. On BG, $g^0 = arg\max_{g \in G} u_5(b^1, g)$ gives the optimal choice in G.
3. message c^0 is sent from BCD to CE. On CE, $e^0 = arg\max_{e \in E} u_2(c^0, e)$ gives the optimal choice in E.
4. message b^1 is sent from BCD to AC. On AC, $a^2 = arg\max_{a \in A} u_1(a, b^1)$ gives the optimal choice in A.

Finally, we get $(a^2, b^1, c^0, d^0, e^1, f^1, g^0, h^4)$ as the optimal solution with utility 44. Remark that optimizing separately each function u_i yields an overall utility of 49 points but this is an infeasible solution.

The whole process is similar to messages sent for computing the most probable explanations in Bayesian networks [Nilsson, 1998]: in the junction tree, a root is first chosen (here BCD), then a collect and a distribute phases are performed from this root. During collect, message sent from a clique C_i to a clique C_j contains the max of the utility over all the variables on C_i's side of the tree except those in $C_i \cap C_j$. During distribution, message from C_j to C_i contains the optimal values of the attributes in $C_i \cap C_j$.

5 Conclusion

In this paper the results obtained are twofold: 1) under a solvability assumption, a general approach for the elicitation of GAI-decomposable utilities over product sets has been proposed; 2) we highlighted the power of GAI networks both in the design of simple elicitation procedures and in the efficient organization of computations for optimization tasks. To go one step further on optimization, one crucial point should be investigated: the case where feasible solutions do not form a whole product set but only a subset implicitly defined by some constraints on the values of the attributes. The procedure we proposed cannot be used directly in this case because it applies only on product set structures. Different solutions can be considered, depending on the relations between the cliques of the constraints graph and those of the GAI-networks. For example, when the former are included in the latter, our procedure can easily be adapted by integrating constraint-violation penalties in the utility functions stored in each clique. In the other cases, we have to solve a general constraint satisfaction problem valued with a GAI-decomposable utility.

References

[Adams, 1965] E Adams. Elements of a theory of inexact measurement. *Philosophy of Science*, 32:205–228, 1965.

[Bacchus and Grove, 1995] F Bacchus and A Grove. Graphical models for preference and utility. In *UAI'95*, 1995.

[Boutilier *et al.*, 2001] C. Boutilier, F. Bacchus, and R. I. Brafman. UCP-networks; a directed graphical representation of conditional utilities. In *UAI'01*, pages 56–64, 2001.

[Boutilier *et al.*, 2004a] C Boutilier, R Brafman, C Domshlak, H Hoos, and D Poole. CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. *Journal of Artificial Intelligence Research*, 21:135–191, 2004.

[Boutilier *et al.*, 2004b] C Boutilier, R Brafman, C Domshlak, H Hoos, and D Poole. Preference-based constraint optimization with CP-nets. *Computational Intelligence*, 20, 2004.

[Brafman *et al.*, 2004] R Brafman, C Domshlak, and T Kogan. On generalized additive value-function decomposition. In *UAI'04*, 2004.

[Chateauneuf and Wakker, 1993] A Chateauneuf and P Wakker. From local to global additive representation. *Journal of Mathematical Economics*, 22:523–545, 1993.

[Cowell *et al.*, 1999] R Cowell, A Dawid, S Lauritzen, and D Spiegelhalter. *Probabilistic Networks and Expert Systems*. Statistics for Engineering and Information Science. Springer-Verlag, 1999.

[Debreu, 1964] G Debreu. Continuity properties of paretian utility. *International Economic Review*, 5:285–293, 1964.

[Fishburn, 1970] P. C. Fishburn. *Utility Theory for Decision Making*. Wiley, NewYork, 1970.

[Gonzales and Perny, 2004] C Gonzales and P Perny. GAI networks for utility elicitation. In *KR'04*, pages 224–234, 2004.

[Gonzales, 2003] C Gonzales. Additive utility without restricted solvability on every component. *Journal of Mathematical Psychology*, 47:47–65, 2003.

[Jensen, 1996] F Jensen. *An introduction to Bayesian Networks*. Taylor and Francis, 1996.

[Krantz *et al.*, 1971] D Krantz, R D Luce, P Suppes, and A Tversky. *Foundations of Measurement (Additive and Polynomial Representations)*, volume 1. Academic Press, 1971.

[Nilsson, 1998] D Nilsson. An efficient algorithm for finding the m most probable configurations in probabilistic expert systems. *Statistics and Computing*, 8(2):159–173, 1998.