Choquet Optimization Using GAI Networks for Multiagent/Multicriteria Decision-Making

Jean-Philippe Dubus, Christophe Gonzales, and Patrice Perny

LIP6 - UPMC
104 avenue du président Kennedy, F-75016 Paris
firstname.lastname@lip6.fr

Abstract. This paper is devoted to preference-based recommendation or configuration in the context of multiagent (or multicriteria) decision making. More precisely, we study the use of decomposable utility functions in the search for Choquet-optimal solutions on combinatorial domains. We consider problems where the alternatives (feasible solutions) are represented as elements of a product set of finite domains and evaluated according to different points of view (agents or criteria) leading to different objectives. Assuming that objectives take the form of GAI-utility functions over attributes, we investigate the use of GAI networks to determine efficiently an element maximizing an overall utility function defined by a Choquet integral.

Keywords: GAI-nets, Choquet Integral, Multiobjective Combinatorial Optimization, Multiagent Decision-Making, Preference-based Configuration.

1 Introduction

The multiplication of preference-based configuration problems has stressed the need for compact preference representation languages and for preference-based optimization algorithms. In this area, graphical models are omnipresent. One can distinguish non-numerical models like CP-nets [1,2] and their extension to the multiagent case mCP-nets [3] on the one hand, and numerical models based on decomposable utility functions like UCP-nets [4] and GAI-nets [5,6,7] on the other hand. In this paper, we investigate the potential of GAI-networks to represent and solve decision making problems where the performance of a solution is evaluated according to different points of view. This type of problem occurs when several criteria, possibly conflicting, must be considered in the decision analysis, or when several agents are involved in the decision process. In both cases, any feasible solution is represented by a vector of utilities. Assuming that each of these utilities is defined by a GAI-decomposable model, we study the use of GAI-nets to determine efficiently a solution having a utility vector maximizing an overall utility function defined by a Choquet integral.

The paper is organized as follows: in Section 2 we explain how GAI-networks are used to represent preferences in multiobjective problems. Then, after recalling

F. Rossi and A. Tsoukis (Eds.): ADT 2009, LNAI 5783, pp. 377–389, 2009.

basic notions linked to capacities and Choquet integrals (Section 3), we introduce a vector-passing algorithm for Choquet-optimization (Section 4). Under the assumption of convex capacity, we propose a refinement of the previous algorithm and a second algorithm based on a ranking procedure (Section 5). Both algorithms have been implemented and tested on randomly drawn instances. The solutions times obtained are given for the sake of comparison (Section 6).

2 GAI Models for Individual and Collective Preferences

In configuration problems, alternatives (feasible solutions) are characterized by n variables (or attributes) $x_1, \ldots, x_n$ taking their values in finite domains $X_1, \ldots, X_n$ respectively. They can thus be seen as elements of the product set of these domains $\mathcal{X} = X_1 \times \cdots \times X_n$. Throughout this paper, by abuse of notation, for any set $\mathbf{Y} \subseteq \{1, ..., n\}$, $X_{\mathbf{Y}}$ refers to $\prod_{i \in \mathbf{Y}} X_i$ and $x_{\mathbf{Y}}$ to the projection of $x \in \mathcal{X}$ on $X_{\mathbf{Y}}$. We also consider preference relations over $\mathcal{X}$ representable by utility functions, i.e., by functions $u : \mathcal{X} \mapsto \mathbb{R}$ such that, for all $x, y \in \mathcal{X}$, $u(x) \geq u(y)$ if and only if x is preferred to y or x and y are judged equivalent. Such functions u are used within solvers to determine the best elements in $\mathcal{X}$ [8].

One major difficulty in using utilities lies in their elicitation: each agent has her own preferences and, hence, her own utility u that needs be constructed prior to being used for optimization tasks. However, on combinatorial domains such as $\mathcal{X}$, elicitation may be impossible as it may involve asking unreasonably large amounts of questions to the agent. Fortunately, it is often the case that subsets of attributes are considered independent by the agent. For instance, the brand of a car may be irrelevant to preferences over its colors, hence inducing an independence between color and brand. In such cases, these independences can be exploited to drastically reduce the elicitation burden. In the literature, different types of independence have been studied such as *preferential independence* or *utility independence* [9,8,10], that induce different decompositions of utility u as a function of subutilities, say u_i's, defined over small sets of attributes.

The most widely used decomposition is the additive one: $u(x) = \sum_{i=1}^{n} u_i(x_i)$ for any $x = (x_1, \ldots, x_n) \in \mathcal{X}$. Note that this model only requires eliciting and storing $u_i(x_i)$ for any $x_i \in X_i$, $i = 1, \ldots, n$. However, such a decomposition is not always appropriate as it inevitably rules out any interaction between attributes, which is far from being realistic. Some generalizations of additive utilities have thus been investigated. In particular, GAI (generalized additive independence) decompositions introduced by [11] are especially attractive as they allow quite general interactions between attributes while preserving some decomposability. Actually, GAI decomposition is a generalization of the additive decomposition in which subutilities u_i's are allowed to be defined over overlapping factors.

Definition 1. *Let* $\mathbf{C}_1, \ldots, \mathbf{C}_k$ *be subsets of* $\mathbf{N} = \{1, \ldots, n\}$ *such that* $\mathbf{N} = \bigcup_{i=1}^{k} \mathbf{C}_i$. *A utility function* $u(\cdot)$ *over* $\mathcal{X}$ *is GAI-decomposable w.r.t. the* $X_{\mathbf{C}_i}$*'s if and only if there exist functions* $u_i : X_{\mathbf{C}_i} \mapsto \mathbb{R}$ *such that:*

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

For instance, $u(a,b,c,d,e,f) = u_1(a,b) + u_2(c,d) + u_3(a,c,e) + u_4(e,f)$ defined on $A \times B \times C \times D \times E \times F$ is a GAI-decomposable utility, with $X_{\mathbf{C}_1} = A \times B$, $X_{\mathbf{C}_2} = C \times D$, $X_{\mathbf{C}_3} = A \times C \times E$ and $X_{\mathbf{C}_4} = E \times F$. GAI decompositions can be represented by graphical structures called *GAI networks* [5]:

Definition 2. *Let $u(x) = \sum_{i=1}^{k} u_i(x_{\mathbf{C}_i})$ be a GAI utility. A GAI net representing u is an undirected graph $\mathcal{G} = (\mathcal{C}, \mathcal{E})$ satisfying the following properties:*

Prop 1: $\mathcal{C} = \{X_{\mathbf{C}_1}, \ldots, X_{\mathbf{C}_k}\}$. *Vertices $X_{\mathbf{C}_i}$'s are called* cliques. *To each vertex $X_{\mathbf{C}_i}$ is associated the corresponding factor u_i from the utility function u;*

Prop. 2: $(X_{\mathbf{C}_i}, X_{\mathbf{C}_j}) \in \mathcal{E} \Rightarrow \mathbf{C}_i \cap \mathbf{C}_j \neq \emptyset$. *Edges $(X_{\mathbf{C}_i}, X_{\mathbf{C}_j})$'s are labeled by $X_{\mathbf{S}_{ij}}$, where $\mathbf{S}_{ij} = \mathbf{C}_i \cap \mathbf{C}_j$. $X_{\mathbf{S}_{ij}}$ is called a* separator*;*

Prop. 3: *for all $X_{\mathbf{C}_i}, X_{\mathbf{C}_j}$ such that $\mathbf{C}_i \cap \mathbf{C}_j = \mathbf{S}_{ij} \neq \emptyset$, there exists a path between $X_{\mathbf{C}_i}$ and $X_{\mathbf{C}_j}$ in $\mathcal{G}$ such that for every clique $X_{\mathbf{C}_h}$ in this path $\mathbf{S}_{ij} \subseteq \mathbf{C}_h$ (running intersection property).*

Cliques are drawn as ellipses and separators as rectangles. For any GAI decomposition, by Definition 2, cliques should be the sets of variables of the subutilities. The edges in the network represent the intersections between subsets of attributes. Fig. 1 shows the GAI net's structure for the example given just below Definition 1. In this paper, we shall only be interested in GAI trees as it is not restrictive [5]. For the elicitation of GAI networks, refer to [5,12,6].

Consider now a finite set of objectives, criteria or agents, $M = \{1, ..., m\}$ and assume that any solution $x \in \mathcal{X}$ is characterized by a utility vector $(u^1(x), \ldots, u^m(x)) \in \mathbb{R}^m$ where $u^i : \mathcal{X} \to \mathbb{R}$ is the i^{th} utility. It measures the relative utility of alternatives with respect to the i^{th} point of view (criterion or agent) considered in the problem. Hence, the comparison of alternatives, say x and y, now reduces to that of their utility vectors $(u^1(x), \ldots, u^m(x))$ and $(u^1(y), \ldots, u^m(y))$.

Each u^i is actually a single utility and, as such, can be GAI decomposable. Assume that all u^i's have the same GAI structure, that is, the u^i's are decomposable as sums of functions u^i_j's whose domains are the same for all i's (but their values differ from one j to another). Then, a GAI net compactly encoding vectors $(u^1, \ldots, u^m)$ can easily be constructed: its graphical structure is that of the GAI net of any u^i (since they are all identical), and each clique $X_{\mathbf{C}_j}$ contains utility vectors $(u^1_j, \ldots, u^m_j)$. Fig. 1 shows how vectors of utilities u^i's decomposable as $u^i_1(a,b) + u^i_2(c,d) + u^i_3(a,c,e) + u^i_4(e,f)$ can be represented by a GAI net. In this figure, tables contain values of utility vectors $(u^1_j, \ldots, u^m_j)$, for fixed j's.

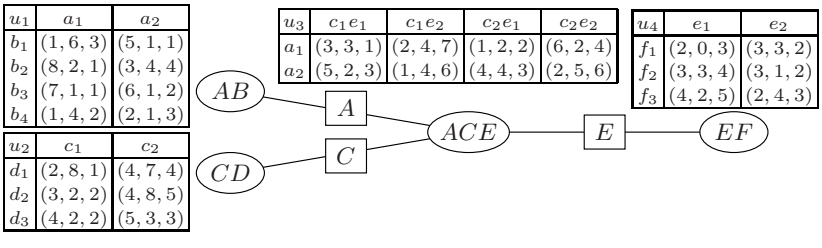

Fig. 1. Example of a GAI network with three criteria

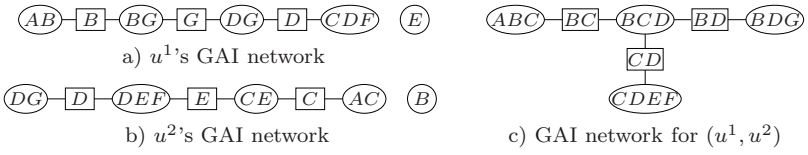

Fig. 2. The GAI trees representing u^1 and u^2 and that for (u^1, u^2)

Of course, in practice, u^i's are seldom decomposable w.r.t. the same GAI structure. For instance, if $\mathcal{X}$ is a set of cars, then, in a family, the utility over $\mathcal{X}$ of the father may be decomposable as $u^1(\text{car}) = u^1_1(\text{price,brand}) + u^1_2(\text{power,speed,consumption}) + u^1_3(\text{speed,security})$ whereas that of the mother may be $u^2(\text{car}) = u^2_1(\text{price,consumption}) + u^2_2(\text{color,brand})$, and the utility of their son $u^3(\text{car}) = u^3_1(\text{brand}) + u^3_2(\text{color}) + u^3_3(\text{power,speed})$. In such a case, we need to find a GAI net with "bigger" cliques that can contain all the u^i_j's functions while encompassing as much as possible the decompositions of the u^i's. For instance, utilities u^1 and u^2 decomposable w.r.t. the GAI nets on the left of Fig. 2 can both be represented (less compactly) by that of Fig. 2.c. Hence vectors (u^1, u^2) are GAI decomposable according to Fig. 2.c. This graph can be constructed by triangulation of the union of the u^i's Markov graphs [13,5].

3 Preference Aggregation with the Choquet Integral

In a multiagent/multicriteria problem, comparing elements of $\mathcal{X}$ amounts to comparing their respective utility profiles. The basic preference model to compare solutions is Pareto dominance defined, for any pair $x, y \in \mathcal{X}$ by: $x \succ_P y \Leftrightarrow u^i(x) \geq u^i(y)$ for all $i \in \{1, \ldots, m\}$ and $u^j(x) > u^j(y)$ for some j. This naturally leads to a primary optimality concept known as Pareto-optimality. Pareto optimal elements in a set $X \subseteq \mathcal{X}$ are those that are Pareto-dominated by no other element in X, i.e., they have a utility profile that cannot be improved on one component without downgrading another one. As shown in [14] Pareto-optimal elements in $\mathcal{X}$ can be computed using vector-valued GAI-networks (see Fig. 1). However, Pareto-dominance is only a partial weak order that leaves many pairs of solutions uncompared. Hence, the Pareto set can be huge due to the combinatorial nature of the problem, and its exact determination requires, for some instances, prohibitive computation times. Fortunately, decision theory provides various preference models refining Pareto dominance and modeling various attitudes in preference aggregation. Among them, the Choquet integral [15] is one of the most expressive decision criteria. It is an aggregation function that generalizes weighted averages when weights are not only attached to each component (criteria or agent) but also possibly to any subset of components. These weights are possibly non additive and are represented by a *capacity* on $M = \{1, \ldots, m\}$.

Definition 3. *A capacity on M is a set function $v : 2^M \to [0,1]$ such that:*
$v(\emptyset) = 0$; $v(M) = 1$; $\forall A, B \in 2^M$ *such that* $A \subseteq B$, $v(A) \leq v(B)$.

For any subset $A \subseteq M$, $v(A)$ represents the importance of coalition A. Let us first recall some definitions about capacities.

Definition 4. *A capacity v is said to be* convex *(or supermodular) when $v(A \cup B) + v(A \cap B) \geq v(A) + v(B)$ for all $A, B \subseteq M$, and it is said to be* concave *(or submodular) when $v(A \cup B) + v(A \cap B) \leq v(A) + v(B)$ for all $A, B \subseteq M$.*

Definition 5. *To any capacity v, we can associate a* dual *capacity $\bar{v}$ defined by $\bar{v}(A) = 1 - v(M \setminus A)$ for all $A \subseteq M$.*

It is well known that $\bar{v}$ is concave if and only if v is convex and vice-versa. Remark that when v is convex, we have $v(A) + v(M \setminus A) \leq 1$, hence $v(A) \leq \bar{v}(A)$. As we shall see later, a useful concept in this case is the core of v defined by:

$$core(v) = \{\lambda \in \mathcal{L} : v(A) \leq \lambda(A) \leq \bar{v}(A)\} ,$$

where $\mathcal{L}$ is the set of probability distributions on M and $\lambda(A) = \sum_{i \in A} \lambda_i$ represents the probability of A. The core is known to be non-empty as soon as v is convex [16]. This result will be used in Section 5. The Choquet integral of a utility vector $u(x) = (u^1(x), \ldots, u^m(x))$ w.r.t. a capacity v is defined by:

$$C_v(u(x)) = \sum_{i=1}^{m} \Big[v(X^{(i)}) - v(X^{(i+1)}) \Big] u^{(i)}(x) = \sum_{i=1}^{m} \Big[u^{(i)}(x) - u^{(i-1)}(x) \Big] v(X^{(i)}) \quad (1)$$

where (.) is a permutation on $\{1, \ldots, m\}$ such that $0 = u^{(0)}(x) \leq u^{(1)}(x) \leq \ldots \leq u^{(m)}(x)$, $X^{(i)} = \{j \in M, u^j(x) \geq u^{(i)}(x)\} = \{(i), (i+1), \ldots, (m)\}$ for $i \leq m$ and $X^{(m+1)} = \emptyset$. Note that $X^{(i+1)} \subset X^{(i)}$, hence $v(X^{(i)}) \geq v(X^{(i+1)})$ for all i. The Choquet integral generalizes averages with the following interpretation based on Eq. (1): for a given utility vector $(u^1(x), \ldots, u^m(x))$, the outcome is at least $u^{(1)}(x)$ with weight $v(X^{(1)}) = 1$, then it increases from $u^{(1)}(x)$ to $u^{(2)}(x)$ with weight $v(X^{(2)})$, then from $u^{(2)}(x)$ to $u^{(3)}(x)$ with weight $v(X^{(3)})$, and so on... The overall integral thus results from aggregation of marginal utility increments $[u^{(i)}(x) - u^{(i-1)}(x)]$ weighted by $v(X^{(i)})$. Note that Choquet Integral includes weighted averages as particular cases. Indeed, when v is additively decomposable, $v(A) = \sum_{i \in A} v_i$ for all $A \subseteq M$, where $v_i = v(\{i\})$. Hence $v(X^{(i)}) - v(X^{(i+1)}) = v^{(i)}$ for all i and $C_v(u(x)) = \sum_{i=1}^{m} v_{(i)} u^{(i)}(x) = \sum_{i=1}^{m} v_i u^i(x)$. When used with a non-additive capacity, it offers enhanced descriptive possibilities.

Example 1. Consider a case with 3 criteria or agents ($M = \{1, 2, 3\}$) and 3 solutions x, y, z with utility vectors $u(x) = (15, 5, 10)$, $u(y) = (10, 10, 10)$ and $u(z) = (5, 15, 10)$ respectively, and the convex capacity v defined in Table 1 (we also give its dual $\bar{v}$ and an additive capacity $p \in core(v)$). $C_v(u(x)) = 5 \times 1 + (10 - 5) \times 0.5 + (15 - 10) \times 0.2 = 8.5$, $C_v(u(y)) = 10 \times 1 = 10$ and $C_v(u(z)) = 5 \times 1 + (10 - 5) \times 0.5 + (15 - 10) \times 0.1 = 8$. Hence according to the model, we get: $y \succ x \succ z$. If we use the dual capacity $\bar{v}$, which is concave, we get $C_{\bar{v}}(u(x)) = 5 \times 1 + (10 - 5) \times 0.9 + (15 - 10) \times 0.5 = 12$, $C_{\bar{v}}(u(y)) = 10 \times 1 = 10$ and $C_{\bar{v}}(u(z)) = 5 \times 1 + (10 - 5) \times 0.8 + (15 - 10) \times 0.5 = 11.5$. Hence with $\bar{v}$

we get: $x \succ z \succ y$. Note that none of the two orders obtained are representable with a weighted sum because $x \succ y$ would imply $y \succ z$, $y \succ x$ would imply $z \succ y$ and conversely. When v is convex we can see that solution y with a flat utility profile is better ranked. On the contrary, with a concave capacity, it seems that solutions x and z with contrasted profiles are preferred to y. As we shall see in Section 5, this is a general feature of Choquet integral: we have to use a convex capacity to exhibit preference for well-balanced solutions and conversely.

In the next section, we investigate the determination of the optimal tuple in $\mathcal{X}$ w.r.t. the Choquet integral. Note that, when choosing $v(A) = 1$ for all non-empty $A \subseteq M$, then $C_v(u(x)) = u^{(m)}(x) = \max_{i \in M} u^i(x)$. Hence the determination of a Choquet-optimal solution reduces to a min-max optimization problem which is known to be NP-hard even when every function u^i is an additive utility [17].

4 A Vector-Passing Algorithm for Choquet Optimization

All GAI message-passing algorithms rely on the same principle: a clique called *root* is chosen to concentrate during a *collect* phase all the information relevant to compute some quantity to be optimized. This phase is processed recursively: *root* asks its neighbor cliques to send it messages containing the aforementioned relevant information; in turn, these neighbors ask their other neighbors to send relevant information, and so on. Once a clique has received all the information it requested, it computes and sends the message it was asked for. Once *root* has received all the information it requested, a *distribute* phase is applied that propagates recursively optimal attributes instantiations from *root* toward the outside of the GAI net. The result of this phase is an optimal instantiation tuple of all the attributes of the GAI network w.r.t. the quantity to be optimized.

As an illustration in the scalar case, consider a utility decomposable according to the graph of Fig. 1: $u(a, b, c, d, e, f) = u_1(a, b) + u_2(c, d) + u_3(a, c, e) + u_4(e, f)$, where each u_i's codomain is $\mathbb{R}$. Assume we wish to find a tuple maximizing u. Let clique EF act as *root*. Collect consists in EF asking ACE to send a message, which in turn asks both AB and CD to send messages. Clique AB sends message $\phi_1(A) = \{\max_b u_1(a, b) : a \in A\}$ and clique CD sends $\phi_2(C) = \{\max_d u_2(c, d) : c \in C\}$, that is, messages $\phi_1(A)$ and $\phi_2(C)$ contain the optimal values of u_1 and u_2 for each value of separators A and C respectively. Then clique ACE sends message $\phi_3(E) = \{\max_{a,c}[u_3(a, c, e) + \phi_1(a) + \phi_2(c)] : e \in E\}$. Finally, *root* EF computes $\max_{e,f}[u_4(e, f) + \phi_3(e)]$, which is the optimal value for u. Actually, as described more formally in [7], we just computed:

$$\max_{e,f}[u_4(e, f) + \max_{a,c}((\max_b u_1(a, b)) + (\max_c u_2(c, d)) + u_3(a, c, e))] = \max_{a,b,c,d,e,f} u \ .$$

The distribute phase just traces back the Argmax's to find the optimal tuple.

In a multiagent/multicriteria setting, where the overall criterion to optimize is a Choquet integral, there is an additional source of complexity: the Choquet integral is not a GAI decomposable function even when u^i's are GAI decomposable. As a consequence, optimality of the solutions cannot be guaranteed by

Table 1. A capacity v for three criteria (named 1,2,3)

A	$\emptyset$	$\{1\}$	$\{2\}$	$\{3\}$	$\{1,2\}$	$\{1,3\}$	$\{2,3\}$	$\{1,2,3\}$
$v(A)$	0	0.2	0.1	0.1	0.4	0.5	0.5	1
$p(A)$	0	0.5	0.4	0.1	0.9	0.6	0.5	1
$\overline{v}(A)$	0	0.5	0.5	0.6	0.9	0.9	0.8	1

passing messages containing only one locally optimal scalar per value of separator (such as $\phi_1(a)$ above). Messages have to carry multiple utility vectors. Indeed, assume we wish finding an optimal tuple w.r.t. a Choquet integral with capacity v defined by Table 1 and utility u over $A \times B \times C$ decomposable as $u_1(a,b) + u_2(b,c)$. For a given value b of B, assume that the message sent by clique AB to BC could be $u_1(a,b) = (3,2,2)$ or $u_1(a',b) = (2,2,4)$. Both utility vectors yield the same Choquet integral: $C_v(3,2,2) = C_v(2,2,4) = 2.2$, hence it is tempting to send only one vector to BC since both vectors seem *a priori* equivalent. However, if the vector received by BC is added to $u_2(b,c) = (1,2,3)$, then $C_v(u_1(a,b)+u_2(b,c)) = 4.1 > C_v(u_1(a',b)+u_2(b,c)) = 3.7$, and if it is added to $u_2(b,c') = (3,2,1)$, then $C_v(u_1(a,b)+u_2(b,c')) = 3.9 < C_v(u_1(a',b)+u_2(b,c')) = 4.4$. As a consequence, locally in clique AB, it is not possible to determine which of $u_1(a,b)$ and $u_1(a',b)$ should be sent to clique BC to determine the optimal solution and we thus need to send both utility vectors on the separator.

Fortunately, not all utility vectors need be sent on separators: for any fixed value of a separator, only Pareto-nondominated vectors need be. As Choquet integral increases with each component, if $x \succsim_P y$, then $C_v(x) \geq C_v(y)$ for any capacity v. Now, once the value of a separator is fixed in a GAI net, it breaks its underlying utility into an additive utility. For instance, in Fig. 1, fixing the value of E to e' decomposes $u(a,b,c,d,e',f)$ into $w_1(a,b,c,d) + w_2(f)$ where $w_1(a,b,c,d) = u_1(a,b) + u_2(c,d) + u_3(a,c,e')$ and $w_2(f) = u_3(e',f)$. Hence, if $w_1(a,b,c,d) \succsim_P w_1(a',b',c',d')$, then adding to both vectors $w_2(f)$ results in $u(a,b,c,d,e',f)$ and $u(a',b',c',d',e',f)$ respectively, the former Pareto dominating the latter. Hence, $C_v(u(a,b,c,d,e',f)) \geq C_v(u(a',b',c',d',e',f))$, and tuple (a',b',c',d',e',f) need not be considered as the optimal tuple. Applying this result on utility u of Fig. 1, if EF is chosen as *root*, only nondominated utilities of message $\mathcal{M}_A$ of Fig. 3 need be sent from clique AB to ACE. Similarly, only non dominated vectors of $\mathcal{M}_C$ need be sent from CD. ACE now needs only send vectors of $\mathcal{M}_E$ to clique EF. Thus, considering messages containing only nondominated utility vectors, we need not examine all the 288 instantiation tuples of $\mathcal{X}$, but we just examine the 8 vectors in u_1 and propagate 4 of them in $\mathcal{M}_A$; we just propagate 4 vectors out of 6 in $\mathcal{M}_C$; clique ACE combines these messages with u_3, thus creating 32 new vectors, of which only 11 are transmitted to clique EF. Finally, EF combines $\mathcal{M}_E$ with u_4, thus creating 33 vectors, and selects that which optimizes c_v, thus highlighting the efficient optimization process.

An additional (global) pruning can be used in conjunction with the above local pruning to speed-up the search: assume that, during our search, we exhibited a complete instantiation x having utility vector $u(x)$. For a new instantiation y to be optimal, $u(y)$ must not be Pareto dominated by $u(x)$. As in the preceding

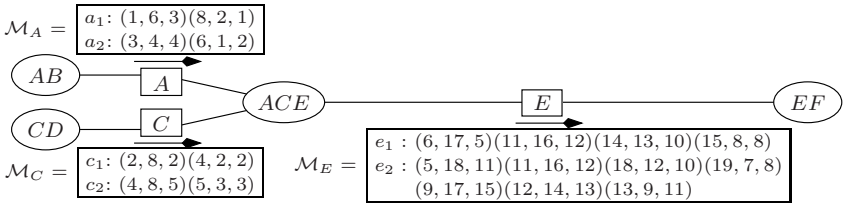

Fig. 3. Sending nondominated messages toward *root EF*

paragraph, assume that the value of separator E is fixed to e', hence decomposing u into $w_1 + w_2$ as described above and assume that we know for sure that, for any $f \in F$, $u_3(e', f) \precsim_P h$ for a given vector h. Then, if $u(x) \succsim_P w_1(a, b, c, d) + h$, no instantiation f is such that $w_1(a, b, c, d) + w_2(f) \succ_P u(x)$ and, thus, $w_1(a, b, c, d)$ needs not be sent on separator E. In this paper, we considered the following heuristic h: given a set of vectors $Z = \{(z_i^1, \ldots, z_i^m), i \in \{1, \ldots, r\}\}$, h is defined as $h = \nabla Z = (z^1, \ldots, z^m)$ where $z^j = \max\{z_1^j, \ldots, z_r^j\}$ for all $j \in \{1, \ldots, m\}$. For clique AB of Fig. 1, $h(a) = \nabla\{u_3(a, c, e) + u_2(c, d) + u_4(e, f)\}$ for all $a \in A$. However, to speed up h's computation, we approximate it by $h'(a) = \nabla\{u_3(a', c, e) + \nabla\{u_2(c', d) : c' = c\} + \nabla\{u_4(e', f) : e' = e\} : a' = a\}$ as follows: first compute $\mathcal{H}_E^D = \nabla\{u_4(e, f)\}$ for all $e \in E$, then $\mathcal{H}_C^C = \nabla\{u_3(c, d)\}$ for all $c \in C$. Finally, compute $u_3 + \mathcal{H}_C^C + \mathcal{H}_E^D$ and apply operator ∇ on it, resulting in $\mathcal{H}_A^D$ (see Fig. 4). The same applies to clique CD: $h'(c) = \nabla\{u_3(a, c', e) + \nabla\{u_1(a', b) : a' = a\} + \nabla\{u_4(e', f) : e' = e\} : c' = c\}$. Here again, there just needs to compute $\mathcal{H}_A^C = \nabla\{u_1(a', b) : a' = a\}$, then $u_3 + \mathcal{H}_A^C + \mathcal{H}_E^D$ and apply operator ∇.

To avoid redundant computations, we can use the following message-passing scheme: let EF act as root. Collect: EF asks ACE to send a message, which in turn asks AB and CD to send messages. AC sends message $\mathcal{H}_A^C = \{\nabla\{u_1(a', b) : a' = a\} : a \in A\}$. Similarly, clique CD sends message $\mathcal{H}_C^C = \{\nabla\{u_3(c', d) : c' = c\} : c \in C\}$. Finally, ACE sends on separator E message $\nabla\{u_3 + \mathcal{H}_A^C + \mathcal{H}_C^C\}$ for each value $e \in E$ (see Fig. 4.a). Distribute phase: EF sends on E message $\mathcal{H}_E^D = \{\nabla\{u_4(e', f) : e' = e\} : e \in E\}$. Clique ACE now computes $\nabla\{u_3 + \mathcal{H}_A^C + \mathcal{H}_E^D\}$ and sends it to CD, and $\nabla\{u_3 + \mathcal{H}_C^C + \mathcal{H}_E^D\}$ and send it to AB. In other words, before sending a message to a neighbor, a clique combines the messages it received from all its other neighbors and, then, applies operator ∇. At the end of the distribute phase, each message $\mathcal{H}^D$ corresponds to heuristic h' (Fig. 4.b).

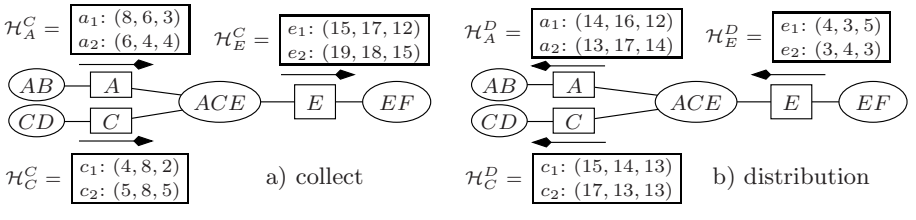

Fig. 4. Propagation of heuristics information

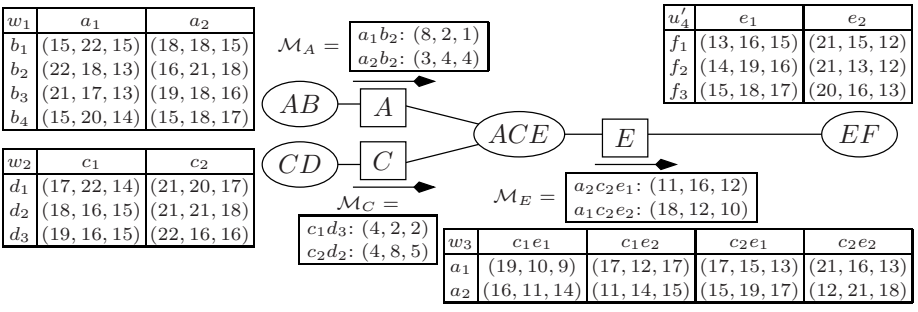

Fig. 5. The vector message-passing algorithm

We can now propose an algorithm in the spirit of MOA* [18] pruning dominated utility vectors, thus reducing the number of vectors sent on separators. First, using h' and a collect, we propagate toward *root* (here EF) on each separator the most promising utility vectors, i.e., those that, given h', have the highest Choquet integrals: clique AB thus sends toward ACE message $\mathcal{M}_A$ containing, for each $a \in A$, the vector $u_1(a, b)$ maximizing $C_v(w_1(a, b))$, where $w_1(a, b) = u_1(a, b) + \mathcal{H}^D_A(a)$. Fig. 5 shows the values of vectors w_1's. In addition, vectors u_1's that are not inserted into $\mathcal{M}_A$ and that are not Pareto dominated by other u_1's are stored into a set of "open vectors" denoted by $\mathcal{L}$ (see Table 2). Set $\mathcal{L}$ thus corresponds to *a priori* less promising vectors that may yet be optimal and, as such, that will need to be sent later on on separators to guarantee the correctness of the algorithm. Similarly, clique CD sends message $\mathcal{M}_C$ containing the u_2's maximizing, for each $c \in C$, $C_v(w_2(c, d))$ where $w_2(c, d) = u_2(c, d) + \mathcal{H}^D_C(c)$. In addition, non dominated u_2's not belonging to $\mathcal{M}_C$ are added to $\mathcal{L}$. Clique ACE now computes vectors $u'_3 = u_3 + \mathcal{M}_A + \mathcal{M}_C$, which correspond to utility of instantiations of attributes A, B, C, D, E. Clique ACE sends in $\mathcal{M}_E$ those u'_3's that maximize, for each value of separator E, $C_v(w_3)$, where $w_3(a, c, e) = u'_3(a, c, e) + \mathcal{H}^D_E(e)$, i.e., the most promising (so far) utility vectors. The other u'_3 vectors are stored into $\mathcal{L}$. Now EF can compute vectors $u'_4(e, f) = u_4(e, f) + \mathcal{M}_E(e)$, which correspond to utilities of complete tuples, and select that which maximizes C_v. Let us call u^* this utility vector. By Fig. 5, $u^* = (15, 18, 17)$ and $C_v(u^*) = 16.1$.

To ensure correctness, we need to send messages of $\mathcal{L}$ toward *root* and check whether they yield better C_v's. But before, we can prune from $\mathcal{L}$ all vectors u_i's such that $C_v(w_i) \leq C_v(u^*)$, since $w_i = u_i + h'$ is an upper bound on the utility of complete tuples compatible with u_i. The set $\mathcal{L}$ of Table 2 thus reduces to the utility vectors of instantiations a_2b_3 and c_2d_3. Note the efficiency of this

Table 2. The set of open vectors $\mathcal{L}$

tuple	u_i, u'_i	$C_v(w_i)$	tuple	u_i, u'_i	$C_v(w_i)$	tuple	u_i, u'_i	$C_v(w_i)$
a_1b_1	(1, 6, 3)	15.7	a_2b_3	(6, 1, 2)	17.2	c_1d_1	(2, 8, 1)	16.0
c_2d_3	(5, 3, 3)	17.2	$a_1c_1e_1$	(15, 7, 4)	11.3	$a_1c_2e_1$	(13, 12, 8)	14.4
$a_2c_1e_1$	(12, 8, 9)	12.9	$a_1c_1e_2$	(14, 8, 10)	13.3	$a_2c_2e_2$	(9, 17, 15)	15.3

pruning rule. We now add the most promising vector of $\mathcal{L}$ to its appropriate separator (here, we can add $u_1(a_2, b_3)$ to $\mathcal{M}_A$) and remove it from $\mathcal{L}$, then clique ACE updates its u'_3 table by computing all the new combinations $u'_3(a_2, c, e) = u_3(a_2, c, e) + u_1(a_2, b_3) + \mathcal{M}_C(c)$. These new combinations are added $\mathcal{L}$, provided that their $C_v(w_3) > C_v(u^*)$ and that they are not Pareto dominated by other u'_3's (for fixed values of separator E). The same process is applied until $\mathcal{L}$ is empty. When a vector from $\mathcal{L}$ is added to the separator adjacent to *root*, the latter updates the value of u^* and prunes $\mathcal{L}$. When $\mathcal{L}$ is empty, all possible optimal combinations have been tested and u^* is an optimal utility vector.

This process is formalized in function `Choquet` below. In this function, we do not use u_i's but rather labels, i.e., triples $\langle v, X_{\mathbf{C}_i}, x_{\mathbf{D}} \rangle$, where v is a utility vector, $X_{\mathbf{C}_i}$ denotes the clique that created the label and $x_{\mathbf{D}}$ is the partial instantiation yielding v. For a given clique $X_{\mathbf{C}_i}$, $\mathit{Labels}(u_i)$ is table u_i in which all vectors are substituted by their label. For a given set of labels $\mathcal{M}$, $\mathcal{M}[x_{\mathbf{E}}]$ denotes the subset of labels $\langle v, X_{\mathbf{C}_i}, y_{\mathbf{D}} \rangle \in \mathcal{M}$ such that $x_{\mathbf{D}\cap\mathbf{E}} = y_{\mathbf{D}\cap\mathbf{E}}$, and $\mathcal{M} \oplus \mathcal{N} = \{\langle v + w, X_{\mathbf{E}}, x_{\mathbf{C}\cup\mathbf{D}} \rangle : \langle v, X_{\mathbf{C}_i}, x_{\mathbf{C}} \rangle \in \mathcal{M}$ and $\langle w, X_{\mathbf{C}_j}, x_{\mathbf{D}} \rangle \in \mathcal{N}\}$. Finally, for any $X_{\mathbf{C}_i}$, we denote $X_{\mathbf{C}_{p(i)}}$ the clique adjacent to $X_{\mathbf{C}_i}$ on the path between `root` and $X_{\mathbf{C}_i}$, and we denote $\mathrm{Adj}(X_{\mathbf{C}_i})$ the set of cliques adjacent to $X_{\mathbf{C}_i}$ except $X_{\mathbf{C}_{p(i)}}$.

Function `Choquet` ()
01 set of open labels $\mathcal{L} \leftarrow \emptyset$; let `root` be any clique
02 **for all** cliques $X_{\mathbf{C}_i}$ from the leaves to the `root` **do**
03 $\mathcal{U} \leftarrow \mathit{Labels}(u_i) \oplus_{X_{\mathbf{C}_k} \in \mathrm{Adj}(X_{\mathbf{C}_i})} \mathcal{M}_k$
04 $\mathcal{M}_i \leftarrow \{$ $\mathcal{U}$'s most promising label for each value of separator $X_{\mathbf{S}_{ip(i)}}\}$
05 $\mathcal{L} \leftarrow \mathcal{L} \cup \{\mathit{ParetoNonDom}(\mathcal{U}[x_{\mathbf{S}_{ip(i)}}] \backslash \mathcal{M}_i[x_{\mathbf{S}_{ip(i)}}])$ for all $x_{\mathbf{S}_{ip(i)}} \in X_{\mathbf{S}_{ip(i)}}\}$
06 send message $\mathcal{M}_i$ on separator $X_{\mathbf{S}_{ip(i)}}$
07 **done**
08 $C_v^{best} \leftarrow \max_{L \in \mathcal{M}_{\mathtt{root}}} C_v(L)$
09 **while** $\mathcal{L} \neq \emptyset$ do
10 let $L = \langle v, X_{\mathbf{C}_i}, x_{\mathbf{D}} \rangle$ be the most promising label in $\mathcal{L}$; remove L from $\mathcal{L}$
11 $\mathcal{M}_i \leftarrow \mathcal{M}_i \cup \{L\}$; $\mathcal{U}'_{p(i)} \leftarrow \{L\} \oplus \mathit{Labels}(u_{p(i)}) \oplus_{X_{\mathbf{C}_k} \in \mathrm{Adj}(X_{\mathbf{C}_{p(i)}}) \backslash \{X_{\mathbf{C}_i}\}} \mathcal{M}_k$
12 **if** $p(i) =$ `root` **then**
13 $C_v^{best} \leftarrow \max\{C_v^{best}, \max_{L \in \mathcal{U}'_{p(i)}} C_v(L)\}$
14 remove from $\mathcal{L}$ labels L's such that $C_v(L \oplus \mathcal{H}^{\mathcal{D}}_{X_{\mathbf{C}_i}}) \leq C_v^{best}$
15 **else**
16 $\mathcal{F} \leftarrow \{L \in \mathcal{L}$ whose clique is $X_{\mathbf{C}_{p(i)}}\}$; $\mathcal{L} \leftarrow (\mathcal{L} \backslash \mathcal{F}) \cup \mathit{ParetoNonDom}(\mathcal{F} \cup \mathcal{U}'_{p(i)})$
17 **done**
18 **return** C_v^{best}

5 The Case of Convex Capacities

Convex capacities are of special interest for Choquet integrals due to their interpretation in terms of preference aggregation: they convey an idea of compromise or fairness named hereafter *preference for well balanced solutions*, meaning intuitively that smoothing or averaging a cost vector improves the alternative. A useful formalization of this idea has been introduced in [19] through an axiom

named *"preference for diversification"* due to its interpretation in the context of portfolio management. This axiom can be reformulated in our framework as:

Definition 6 (Preference for well-balanced solutions). *Preference for well-balanced solutions holds for a relation $\succsim$ on $\mathcal{X}$ if, for any n utility vectors $u_1, \ldots, u_n \in \mathbb{R}$, and for all real numbers $\alpha_1, \ldots, \alpha_n \geq 0$ such that $\sum_{i=1}^{n} \alpha_i = 1$:*

$$[u_1 \sim u_2 \sim \ldots \sim u_n] \Longrightarrow \sum_{i=1}^{n} \alpha_i u_i \succsim u_k, \quad k = 1, \ldots, n \ .$$

When using a Choquet Integral, the above axiom is equivalent to choosing a convex capacity v as shown in [19]. Coming back to Example 1, we can imagine a fourth solution w with utility vectors $u(w) = (12.5, 5, 10)$ such that $C_v(u(w)) = 5 \times 1 + (10 - 5) \times 0.5 + (12.5 - 10) \times 0.2 = 8$. Now, remarking that $C_v(u(z)) = C_v(u(w)) = 8$, preference for well-balanced solutions induced by the convexity of v implies that vector $0.5u(z)+0.5u(w) = (9.75, 10, 10)$ would be preferred to $u(z)$ and $u(w)$. Observing that $u(y)$ Pareto-dominates $u(w)$, we deduce that $u(y)$ is also preferred to $u(z)$ and $u(w)$ by monotonicity of the Choquet integral. Hence resorting to a convex capacity might be natural in many decision situations where a compromise is sought among conflicting points of view. Let us show now how the convexity of v can be exploited on the algorithmic side.

Let $C_v(u(x))$ be the value of the Choquet integral for any x, and let $\bar{u} : \mathcal{X} \mapsto \mathbb{R}$ be a convex combination of marginal utilities defined by: $\bar{u}(x) = \sum_{i=1}^{m} p_i u^i(x)$ where p is a probability distribution in $core(v)$. Such a distribution can be determined using a greedy algorithm [20] (such a p is given in Table 1). Then the following property holds: $\bar{u}(x) \geq C_v(u(x))$ for all $x \in \mathcal{X}$ [21,22]. Hence $\bar{u}(x)$ is an upper bound for the Choquet integral. This can be exploited in the algorithm of Section 4 in conjunction with the pruning rule of heuristic h': after scalarizing the tables of utility vectors of Fig. 1 by $\bar{u}_i(x) = \sum_{j=1}^{m} p_j u_i^j(x)$, applying the same process that enabled us to compute $\mathcal{H}^D$ with these new tables yields a scalar heuristic $\mathcal{H}^p$ stored on each separator. $\mathcal{H}^p$ can be used to prune any utility vector u'_j to be sent by clique $X_{\mathbf{C}_j}$ on a separator whenever $\bar{u}_j + \mathcal{H}^p_j \leq C_v(u^*)$.

Utility $\bar{u}$ can also be used directly for computing a Choquet-optimal element. As a linear combination of GAI utilities, $\bar{u}$ is also a GAI function. Hence, we can rank efficiently elements of $\mathcal{X}$ by decreasing value of $\bar{u}(x)$ [17]. Now, assume that $x^1, ..., x^k$, the k-best elements on $\mathcal{X}$ w.r.t. $\bar{u}$, have been computed. Let $\hat{x}^k = \text{argmax}_{i=1,\ldots,k} C_v(u(x^i))$. If $C_v(u(\hat{x}^k)) \geq \bar{u}(x^k)$, then $\hat{x}^k$ is the optimal choice for C_v. Indeed, as we rank elements w.r.t. $\bar{u}$, for any $k' > k$, $\bar{u}(x^{k'}) \leq \bar{u}(x^k)$, and since $\bar{u}(x) \geq C_v(u(x))$ for all $x \in \mathcal{X}$, $C_v(u(x^{k'})) \leq \bar{u}(x^{k'}) \leq \bar{u}(x^k) \leq C_v(u(\hat{x}^k))$. Consequently, the optimal choice for C_v can be obtained by ranking elements x^i w.r.t. $\bar{u}$ until the maximum value of the $C_v(u(x^j))$'s for all the x^j's found so far, exceeds $\bar{u}(x^i)$. This is another way of generating C_v optimal elements in $\mathcal{X}$ but, contrary to algorithm presented in Section 4, this algorithm is only feasible for a capacity with a non-empty core. This is obviously the case when v is convex.

Table 3. Response times for Choquet optimization

n	$m=2$ VPA*	$m=5$ VPA*
5	0.004	0.015
10	0.06	34.27
15	13.62	467.35

Table 3.a

n	$m=2$, v convex VPA*	VPA*+S	Rank
5	0.004	0.003	0.007
10	0.06	0.05	0.47
15	11.30	6.55	>1200

Table 3.b

n	$m=5$, v convex VPA*	VPA*+S	Rank
5	0.012	0.008	0.009
10	33.34	0.28	9.62
15	451.84	443.76	>1200

Table 3.c

6 Experimentations

In order to evaluate in practice the performance of our algorithms, we compared the vector-passing algorithm of Section 4 (hereafter denoted VPA*), VPA* with heuristic $\mathcal{H}^P$ (denoted VPA*+S) and the ranking algorithm mentioned above (Rank). In each experimentation, X_i's domains were all of size 5. GAI networks were randomly generated with an average of 3.5 attributes per clique. Utility tables were filled with numbers drawn randomly between 0 and 100. All experiments were performed on a 2.13GHz PC with 3GB of RAM. The tables below report average response times in seconds over 2000 experiments. For Table 3.a, we generated random capacities (not necessarily convex). VPA* reveals efficient for instances where the ranking approach does not apply (non-convex cases). Tables 3.b and 3.c are given for the sake of comparison of the two variants of VPA* with Rank in the convex case. Rank is known to be very efficient when criteria are positively correlated. We deliberately generated instances with conflicting (negatively correlated) criteria to check whether VPA* was able to outperform Rank in this case. The answer is clearly positive considering Tables 3.b and 3.c.

References

1. Domshlak, C., Brafman, R.: CP-nets: Reasoning and consistency testing. In: Proc. of KR (2002)
2. Boutilier, C., Brafman, R., Domshlak, C., Hoos, H., Poole, D.: CP-nets: A tool for representing and reasoning with conditional ceteris paribus preference statements. Journal of Artificial Intelligence Research 21, 135–191 (2004)
3. Rossi, F., Venable, K.B., Walsh, T.: mCP nets: Representing and reasoning with preferences of multiple agents. In: Proc. of AAAI, pp. 729–734 (2004)
4. Boutilier, C., Bacchus, F., Brafman, R.: UCP-networks; a directed graphical representation of conditional utilities. In: Proc. of UAI (2001)
5. Gonzales, C., Perny, P.: GAI networks for utility elicitation. In: KR (2004)
6. Braziunas, D., Boutilier, C.: Local utility elicitation in GAI models. In: Proc. of UAI (2005)
7. Gonzales, C., Perny, P., Queiroz, S.: GAI networks: Optimization, ranking and collective choice in combinatorial domains. Foundations of computing and decision sciences 32(4), 3–24 (2008)
8. Keeney, R.L., Raiffa, H.: Decisions with Multiple Objectives - Preferences and Value Tradeoffs. Cambridge University Press, Cambridge (1993)
9. Krantz, D., Luce, R.D., Suppes, P., Tversky, A.: Foundations of Measurement (Additive and Polynomial Representations), vol. 1. Academic Press, London (1971)

10. Bacchus, F., Grove, A.: Graphical models for preference and utility. In: Proc. of UAI (1995)
11. Fishburn, P.C.: Interdependence and additivity in multivariate, unidimensional expected utility theory. International Economic Review 8, 335–342 (1967)
12. Gonzales, C., Perny, P.: GAI networks for decision making under certainty. In: IJCAI 2005 – Workshop on Advances in Preference Handling (2005)
13. Cowell, R., Dawid, A., Lauritzen, S., Spiegelhalter, D.: Probabilistic Networks and Expert Systems. Stats for Engineering and Information Science. Springer, Heidelberg (1999)
14. Dubus, J.P., Gonzales, C., Perny, P.: Multiobjective optimization using GAI models. In: Proc. of IJCAI (2009)
15. Choquet, G.: Theory of capacities. Annales de l'Institut Fourier 5, 131–295 (1953)
16. Shapley, L.: Cores of convex games. Int. J. of Game Theory 1, 11–22 (1971)
17. Gonzales, C., Perny, P., Queiroz, S.: Preference aggregation with graphical utility models. In: Proc. of AAAI, pp. 1037–1042 (2008)
18. Stewart, B.S., White III, C.C.: Multiobjective A*. J. ACM 38(4), 775–814 (1991)
19. Chateauneuf, A., Tallon, J.M.: Diversification, convex preferences and non-empty core in the choquet expected utlity model. Economic Theory 19, 509–523 (2002)
20. Jaffray, J.Y.: On the maximum probability which is consistent with a convex capacity. Int. J. of Uncertainty, Fuzziness and KB Systems 3(1), 27–34 (1995)
21. Galand, L., Perny, P.: Search for choquet-optimal paths under uncertainty. In: Proc. of UAI, pp. 125–132 (2007)
22. Queiroz, S.: Multiperson Choquet-compromise search on large combinatorial domains. In: 2nd IEEE Int. Workshop on Soft Comp. Applications, pp. 187–192 (2007)