Cooperative Agents based Multicriteria Decision Aid

Ins Ben Jaafar & Khaled Ghdira
Department of computer science
High Institute of management
41, rue de la libert, 2000 cit Bouchoucha.
TUNISIE

Abstract: - Decision making consists in choosing, on the basis of various criteria, an appropriate subset of actions among a set of alternatives. The most research related to this field proceeds by aggregating all the criteria in a single objective function and rank alternatives according to this scalar measure. Unfortunately, this is often quite inadequate because it risks to alter the final decision. To discard this drawback, outranking based centralized methods have been proposed. However, they remain insufficient because they don't match with the logical distribution of criteria. That is why we propose a distributed approach which finds out the best compromise between all criteria by considering them as cooperative agents. The underlying foundations are detailed and illustrated via both an example and experimentation.

Key-Words: - Multicriteria optimization, decision making, cooperative agents, multi-agent systems.

1 Introduction

Specialists in multicriteria decision aid aim at giving the decision-maker some tools, in order to deal efficiently with decision problems where several-often contradictory-points of views must be taken into account. The first family of tools consists in aggregating the different points of view into a unique function which must subsequently be optimized. Unfortunately this is very quite inadequate, because the criteria generally are incommensurable. They measure different properties that cannot be related to each other directly and cannot be combined into a single function. Indeed such scheme reflects neither real optimization nor the expected result from all criteria points of view[3,13]. The second family aims firstly to build a relation, called an outranking relation, which represents the decision-maker's strongly established preferences and secondly to exploit the outranking relation in order to help the decision-maker solve his problem[3,13]. Furthermore preference modeling and defining the set of decisions are indispensable and complex steps which have been at the center of some research. The third and most recent family proposes methods which alternate calculation steps and dialogue steps with the decision-maker in order to select appropriate and pertinent information on his preferences [13]. Despite the varieties of research in this field, the obtained results remain insufficient. We think that is may be due to the fact that all these methods are centralized. However, the notion of multicriteria decision problem is logically distributed. That is why we propose to explore distributed artificial intelligence and more specifically the multi agent systems that focus on cooperative and distributed problem solving techniques [2,4,5].

The main objective of this paper is to introduce a Multi Agent model for dealing with multicriteria decision-aid, called Cooperative Agents based Multicriteria Decision-Aid "CAMDA" where each agent is responsible for a single criterion. Thus, the criteria are separately optimized without any "scalarization" form. The entities, called Criterion Agents, cooperate and negotiate in order to find out a sub-set of solutions, compromise sub-set.

CAMDA is detailed in the second section, illustrated through an example in the third section and experimented in the fourth one. The last section contains concluding remarks and future work.

2 The foundations of CAMDA model

CAMDA consists of agents that cooperate to find out a compromise-set of solutions rather than Pareto optimal solutions because Pareto optimality alone is
not always adequate for pin-pointing the final
decision. The set of Pareto optimal choices is often
very large and even after eliminating all the
alternatives that are not Pareto optimal, the user is
usually left with a large number of alternatives to
choose from[3]. In the other hand, the compromise
set consists of solutions obtained by means of
cooperation and negotiation between all criteria.
This set is limited, it’s size is controlled by the user.
Before detailing CAMDA, let us recall some useful
definitions.

Definition1: A multicriteria decision problem is a
situation in which, having defined a set A of actions
and a consistent family F of criteria on A, one
wishes to:

\[
\min_{x \in A} F(x) = \begin{cases} 
  f_1(x) \\
  f_2(x) \\
  \vdots \\
  f_n(x)
\end{cases}, \quad n \geq 2 
\]

Where \( A \) denotes the feasible set of design
alternatives, or the design space[10].

Definition2: The vector \( F(U) \) is said to dominate
another vector \( F(U') \), denoted \( F(U) \prec F(U') \), if and
only if \( f_i(U) \leq f_i(U') \) for all \( i \in \{1, 2, \ldots, n\} \) and
\( f_j(U') < f_j(U) \) for some \( j \in \{1, 2, \ldots, n\} \). A point \( U^* \in A \)
is said to be Pareto optimal or an efficient point for
\((MOP)\) if and only if there does not exist \( U \in A \)
 satisfying \( F(U) < F(U^*) \). The vector \( F(U^*) \) is then
called non-dominated or non inferior.

The above definition qualifies Pareto optimality in
the global sense[10,13]. The definition of local
Pareto optimality is similar and can be found in [3].

2.1 The architecture of CAMDA model
CAMDA benefits from the multi-agent
techniques[2,4,5]that have opened a natural and
efficient way to solve diverse problems in terms of
cooperation, conflict and concurrence within a
society of agents. Each agent is an autonomous
entity that is asynchronously able to acquire
information from its environment and/or from other
agents, to reason on the basis of this information
and to act consequently. Within the large domain of
multi-agent systems, our model consists of
cognitive agents whose number is often limited ( here
equal to the total number of criteria). The multi-agent architecture consists of Criterion agents
cooperating in order to select a sub-set of solutions
corresponding to the best compromise from the
global satisfaction points of view. However, this
class is not sufficient: an Interface is necessary
between this society of agents and the user
essentially to detect that the problem is solved.
Consequently, a second and last class, called
Interface, was created. It contains a single
individual (figure1).

Consider a Criterion agent \( C_{ri} \). It’s static
knowledge corresponds to the criterion to be
optimized \( f_i \) ( figure1), the set of actions \( A \) and the
cost of each action with regards to \( f_i \) and its ideal
action valuation denoted \( \text{id}_i \). Note that the ideal
action corresponds to the best action according to \( f_i \).
Its dynamic knowledge consists of its current
satisfaction level denoted \( s_{li} \) expressing the
satisfactory utility level for \( f_i \), its current frequency
denoted \( f_{Cr_i} \) expressing how much time the set of
action \( A \) has been modified, and its anti-ideal
solution valuation denoted \( \text{aid}_i \) which corresponds
to the worst action according to \( f_i \). Note that
criterion acquaintances consist of the set of all the
other Criterion agents denoted \( Aq_i \).

The Interface Agent has as acquaintances all the
Criterion agents. Its static knowledge consists of the
set of actions \( A \), whereas its dynamic one consists of
the Criterion frequencies and the final result.

Fig.1. Example of CAMDA architecture with 3 criteria

2.2 The dynamic of CAMDA model
It consists of two phases: dynamic Initialization and
Cooperation.
At the first phase, the Interface agent creates \( n \)
Criterion agents and associates to each one of them
a criterion \( f_i \) (figure1). Then, each Criterion \( C_{ri} \)
computes, on the one hand, the cost of each action \( x \)
belonging to the set of actions, \( A \), and on the other
hand, both its ideal id, and anti-ideal aid, costs as follows:

\[ aid = \min(f_i(x)) \text{ (for maximization), } x \in A \]
\[ id = \max(f_i(x)) \text{ (for maximization), } x \in A \]

In the second phase, each Criterion agent Cr_i dynamically computes its satisfaction level sl_i as follows:

\[ sl_i = aid_i + (id_i - aid_i) \epsilon \]

\[ \epsilon \text{ is an adjustment parameter in } [0,1] \text{ used to gradually increase } sl_i \text{ (algorithm1. line5) and consequently to reduce the search space namely the set of actions } A_i \text{ into } A_i \text{ (algorithm1. line6). If } A_i \text{ is empty, i.e. the search space has been too much compressed, then Cr_i backtracks and tries to expand it by modifying } sl_i \text{ as follows: } \]

\[ sl_i = sl_i - (id_i - sl_i) \epsilon / 2 \]

This process is repeated until A_i is not empty (algorithm1. lines8-10). Hence, each Criterion agent Cr_i extracts a sub-set A_i from A_i. Each solution in A_i has a cost greater than sl_i (in the case of maximization that we adopt here) or lower than sl_i (in the case of minimization). Then A_i is sent to the others criteria Cr_j to be checked (algorithm1. line11).

When receiving A_i, each Criterion Cr_j selects from A_i the sub-set of solutions A_jI where cost f_j is in [aid_j, id_j] and then answers Cr_i by sending it A_jI (algorithm1. line24). As the expected answers are received, Cr_i computes the intersection of the sub-sets A_jI j=1..|A_qi| (algorithm1. line26). Once Cr_i has received all the answers, it activates the Adjustment process. Three cases may occur:

- \[ 0 \neq \text{card} (In) \leq k \]
- \[ 0 = \text{card} (In) \]

Given that, the process has reached a sufficiently good compromise namely the best equilibrium state relative to the predefined k value. Thus, Cr_i communicates the final result to the Interface that informs the user and interrupts the search process.

- \[ 0\neq \text{card} (In) \leq k \]

Given that, the agent Cr_i chooses the agent Cr_j having the maximal acceptance ratio corresponding to the size of \( A_i - A_jI \) divided by fCr_j and asks it to expand its current sub-space. When more than one agent have the same refusal ratio, Cr_i chooses randomly one of them. While performing a "Expand-sub-space" message (algorithm1. lines12-19), Cr_j ignores all the future similar incoming messages. Whenever a criterion Cr_i is requested to modify its current set of action A_i, it increments its modification frequency fCr_j and communicates it to the other acquaintances.

Note that the less is k , the best is the quality of the compromise-set.

3. Illustrative example

Consider a consumer who wants to buy a television set and, after a first selection, retains eight models and then evaluates them by taking into account the price, quality of picture (PQ), quality of sound (SQ) and maintenance contract (MC) (V.G.= good, G.= good, A.= average, N.G. = not good). (V.G was somewhat arbitrarily replaced by 2, G. by 1, A. by and N. G by -1) [ 13].

The data are represented by the table1 where we have modified the prices by changing signs, thereby getting a maximization problem.

<table>
<thead>
<tr>
<th>Model</th>
<th>- Price</th>
<th>P.Q</th>
<th>S.Q</th>
<th>MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>-1300</td>
<td>V.G/2</td>
<td>V.G/2</td>
<td>A/0</td>
</tr>
<tr>
<td>T2</td>
<td>-1200</td>
<td>V.G/2</td>
<td>V.G/2</td>
<td>G/1</td>
</tr>
<tr>
<td>T3</td>
<td>-1150</td>
<td>V.G/2</td>
<td>G/1</td>
<td>G/1</td>
</tr>
<tr>
<td>T4</td>
<td>-1000</td>
<td>G/1</td>
<td>G/1</td>
<td>N.G/-1</td>
</tr>
<tr>
<td>T5</td>
<td>-950</td>
<td>G/1</td>
<td>G/1</td>
<td>A/0</td>
</tr>
<tr>
<td>T6</td>
<td>-950</td>
<td>A/0</td>
<td>G/1</td>
<td>N.G/-1</td>
</tr>
<tr>
<td>T7</td>
<td>-900</td>
<td>G/1</td>
<td>A/0</td>
<td>N.G/-1</td>
</tr>
<tr>
<td>T8</td>
<td>-900</td>
<td>A/0</td>
<td>A/0</td>
<td>A/0</td>
</tr>
</tbody>
</table>

Table1: Data example
While not equilibrium state do
1. if mailBox not empty then m— getMsg (m)
2. case m
3. reduce-sub-space (In):
4. aid ← generate-anti-ideal-cost (In)
5. sl ← aid + (id. aid) * ε
6. A ← modification-set (In, sl)
7. f0v ← f0v + 1
8. while (i is empty) do
9. sl ← sl - ((id, sl) * ε/2)
10. if nbr-answers = 0 then
11. sendMsg (Aq i, Cr i, "recommend-solving")
12. expand-sub-space : sl ← sl - ((id, sl) * ε/2)
13. f0v ← f0v + 1
14. while (i is empty) do
15. sl ← sl - ((id, sl) * ε/2)
16. if modfication-set (A, sl)
17. sendMsg (Aq i, Cr i, "request-checking-solutions")
18. request-checking-solutions (Aj):
19. for each solution xi ∈ A do
20. if Sl (Aj) ∈ S then id Aj ∪ {xi}
21. sendMsg (C, C, "answer-checking")
22. answers-checking-solutions (Aj)
23. In — In \ Ai /Aj
24. count-answers
25. if nbr-answers = |Aq| then Adjustment (In)

Algorithm 1: Behavior of Criterion agent Cr i

The dominance relation introduced in the definition can be represented by the accompanying graph of figure 2:

![Graph](image)

Fig. 2. The dominance relation

Note that if there is an indifference threshold of 50 on the prices, one could be tamped to say that T2 dominates T3.

The efficient actions are T2, T3, T5, T7 and T8.

The compromise-set of actions generated by the CAMDA is presented through the different steps of table 2. We take as numerical values ε=20% and K=5.

<table>
<thead>
<tr>
<th>aid</th>
<th>id</th>
<th>sl</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cr1</td>
<td>-900</td>
<td>1220</td>
</tr>
<tr>
<td>Cr2</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Cr3</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>Cr4</td>
<td>-1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: Different steps of CAMDA dynamic

For simplification, the interactions between the different Criterion agents have been omitted namely successive solutions checking. For each agent we have presented only the set of actions \( A_i \) satisfying its level \( sl_i \) and the global intersection (In). Thus, each Criterion agent Cr i extracts a sub-set \( A_i \) from \( A \). Then \( A_i \) is sent to the other criteria Cr j to be checked and this via the message "request-checking-solutions \((A_i/j)\)". When receiving \( A_i \) for example, each Criterion Cr j selects from \( A_i \) the sub-set of solutions \( A/j \) which cost \( f_j \) is in \([aid, id]\). So, Cr2 determines \( A/j:={T2, T3, T4, T5, T7} \) and send it to Cr1 agent via the message "answer-checking-solutions \((A/j)\)". Similarly for the other criteria agents, the following sub-set of solutions \( A/j:={T2, T3, T4, T5, T7} \) are sent to Cr1. As the expected answers are received, Cr1 computes the intersection (In) of the sub-sets \( A/j j=1...[Aq]\). The size of the intersection set \((In)\) is lower than \( K \), so the decision process stops and the compromise-set of solutions \({T2, T3, T4}\) is obtained.

4 Experiments

4.1 Experiment design

The experiments are based on randomly generated binary CSMPs (Constraint Satisfaction and Multicriteria Optimization Problems)[1]. Constraint Satisfaction and Multicriteria Optimization Problem CSMP is inspired from Constraint Satisfaction Problems "CSP" [6,7,8,11] that only focus on constraint satisfaction. It’s a formalism that consists of variables associated with their domains, constraints involving subsets of variables and a set of c functions where each function is a performance criterion mapping every solution to a numerical value. The generation is guided by classical CSP parameters[9,12]: number of variables \((n)\), domain size \((d)\), constraint density \(p\) (a number between 0 and 100% indicating the ratio between the number of the problem effective constraints to the number of all possible constraints, i.e., a complete constraint graph), constraint tightness \(q\) (a number between 0 and 100% indicating the ratio between the number of couples of values forbidden (not allowed) by the constraint to the size of the domain cross product) and the number of performance criterion \((ncr)\).

As numerical values, we use \( n = 10, d = 6, p \) (resp. \( q \)) varying from 50% to 65% (resp. 35% to 50%) by steps of 5% to both keep the solutions number reasonable and to guarantee consistent problems. The adjustment parameter \( \varepsilon \) varies in
{10%, 20%, 30%, 40%, 50%} whereas ncr in {4, 6, 8}. Regarding k, it is set to 10%k₀ where k₀ is equal to the initial size of the set actions A.

4.2 Evaluation parameters
CAMDA is assessed through two families of measures: efficiency and quality. In terms of efficiency, we measure the run time requested to reach an equilibrium state. Concerning the quality, it is quantified as follows:
Suppose that the compromise-set is denoted by Cs
∀ xᵢ ∈ Cs, dᵢ = ∑ᵢ=1..c |idᵢ| fⱼ(xᵢ)/idᵢ, where idᵢ is the ideal cost and fⱼ(xᵢ) is the cost of xᵢ according to the Criterion j.
Let us define the quality by d such that d = infᵢ=1..|Cs| (dᵢ).
So, the lower is d, the better is the quality. This procedure is used just to assess our model and to yield objectively best solutions.
Thus, we have generated 240 examples corresponding each one to the configuration (p, q, ncr, ε). Due to the non-deterministic character of our model, we have performed each example 10 times and then we have taken the average for all parameters.

4.3 Experimental results
Run time versus both tightness and Criterion number (figure3)
As the tightness decreases, the run time increases. This phenomenon can be explained by the fact that the number of solutions grows at low tightness and then the decision process spends too much effort to reach the equilibrium state.
For example, for (p=0.6, q=0.35 and ncr=4) the solution number is equal to 198 whereas it is equal to 15 for (p=0.6, q=0.5 and ncr =4).

Run time versus both density and Criterion number (figure4)
As the density increases, the run time decreases. In fact, the more the density is, the less the solutions number is and then the equilibrium state is quickly reached. For example, for (p=0.5, q=0.5, ncr =4) the solution number is equal to 40 whereas it is equal to 10 for (p=0.6, q=0.5, ncr =4). Notice that for a given couple (p,q), the maximal run time often increases with the Criterion number. Figure 4 presents an almost linear tendency whereas figure 3 gives heterogeneous results. These ones may be explained by the inter-Criteria conflict effect.

Which adjustment parameter for which inter-Criteria conflict?
As before mentioned, the inter-Criteria conflict seems to be important in the decision phase. That is why we perform the following experiments. In fact the aim is to affect the right adjustment parameter ε to the right inter-Criteria conflict parameter.
Let us call this later conflict rate τ and define it by the conflictive criterion number divided by the total number of criteria.
So, let us vary τ in {0.2, 0.4, 0.6, 0.8} and ε in {10%, 20%, 30%, 40%}. Table 3 reports the sum of all Criterion frequencies denoted F (F=Σᵢ=1..ncr fCrᵢ), whereas the table 4 reports the sum of all Criterion backtracks denoted B (B=Σᵢ=1..ncr backtracks).

<table>
<thead>
<tr>
<th>ε</th>
<th>τ→</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>43</td>
<td>11.4</td>
<td>10</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>18</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30%</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40%</td>
<td>15.75</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 3. The sum of all criterion frequencies F

<table>
<thead>
<tr>
<th>ε</th>
<th>τ→</th>
<th>0.2</th>
<th>0.4</th>
<th>0.6</th>
<th>0.8</th>
</tr>
</thead>
<tbody>
<tr>
<td>10%</td>
<td>20</td>
<td>0.4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20%</td>
<td>6.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30%</td>
<td>00</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40%</td>
<td>05.75</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4. The sum of all criterion backtracks B
Note that $F$ and $B$ reflect the optimization effort. So, the greater they are and the worst the run time is. Consequently, we select, from Table 3 and 4, for each $\tau$ the best $\varepsilon$, i.e., the $\varepsilon$ that provides the minimal values of $B$, $F$ and $d$.

-Slightly inter-criteria conflict
When $\tau$ = 0.2, the Criteria are slightly conflictive.
Table 3 and 4 shows that $F$ and $B$ decreases as $\varepsilon$ increases. This can be explained by the fact that when $\varepsilon=10\%$ or 20%, the intersection is too dense and must be compressed until it satisfies the condition "card(In)\leq k" which increases $F$ and $B$. So, it is more appropriate to choose $\varepsilon=30\%$ or 40%.
Concerning $d$, the experimental results show that it doesn't change.

-Highly inter-criteria conflict
When $\tau$ takes value in {0.4,0.6,0.8}, the Criteria are more conflictive then before. Table 3 and 4 shows that $F$ and $B$ decrease as $\varepsilon$ decreases. This can be explained by the fact that when $\varepsilon=10\%$ or 20% the solutions corresponding to the different criteria are located in dispersed regions as the criteria are very conflictive. Consequently, one had better choose little $\varepsilon$ values, otherwise the intersection may be empty which requires more frequencies and backtracks to both reach an nonempty intersection and the equilibrium state. Concerning $d$, the experimental results also show that it is almost the same.

These results show that the adjustment should be small when the criteria are conflictive, otherwise it should be large. Moreover, other experimental results performed show that the quality, i.e. $d$ is as worst as $\tau$ increases.

5 Conclusion and future work
In this paper, we have developed an agent based model for dealing with multicriteria decision-aid. In this model each agent is responsible for a single criterion. Thus the criteria are separately optimized without any "scalarization" form by considering them as cooperative agents trying to reach their best equilibrium state which corresponds to the best compromise-set. The effectiveness of the model is demonstrated in the domain of Constraint Satisfaction and Optimization Problems and discussed on randomly generated examples. The experiments have shown, on the one hand, that as the tightness (density) decreases the optimization effort increases, and on the other hand, that the run time often increases with the criterion number.
Moreover, they have provided that the quality gets worse as the inter-criteria conflict increases.
As far as our future work is concerned, other experiments will be performed. In addition, we shall extend our model to the dynamic aspect that concerns a restriction (a criterion is removed from the MOP) and/or a relaxation (a new criterion is imposed).

References: