Advanced Studies in Contemporary Mathematics
29 (2019), No. 1, pp. 125 - 146

www.jangjeon.or.kr
http://dx.doi.org/10.17777/ascm2019.29.1.125

A HYBRID METHOD TO SOLVE THE MULTI-OBJECTIVE COMBINATORIAL AUCTIONS

CHAHRAZAD ADICHE AND MÉZIANE AÏDER

Abstract. In this paper, we present a hybrid method for solving the multi-item multi-unit Winner Determination Problem of Combinatorial Auctions in the multi-attribute (multi-objective) context. Indeed, the bids may concern several specifications of the item, involving not only its price, but also its quality, the delivery conditions, the delivery deadlines, the risk of not being paid after a bid has been accepted and so on. The problem is intractable and is equivalent to a Multi-Objective Multi-Constraint Knapsack Problem, a well known NP-Hard Problem. We propose a hybrid method, based on the Multi-Objective Branch-and-Bound approach and the Random Walk Tabu Search metaheuristic. The Multi-Objective Branch-and-Bound used here is referred to be the process of the principal research. We present a novel rule to automatically rank bids while taking into account the Decision Makers's (DM's) preferences on objectives that are most relevant. A fuzzy dominance relation, on the discrete set of weight vectors, is then computed and used to rank bids and select a feasible solution (a subset of accepted bids). Numerical experiments are reported on data sets available in the literature, in the case of three objective functions, three items and the number of bids varying from 10 to 50. The obtained results show the efficiency of our Extended Multi-Objective Branch-and-Bound method that outperforms the existing Multi-Objective Branch-and-Bound methods both in terms of CPU time and ratio of dominated partial solutions. Furthermore, the hybrid method generates a larger number of efficient bids in reasonable time for all instances.

2000 Mathematics Subject Classification. 90C29, 90C27, 90C59.

Keywords and phrases. Winner determination problem, Combinatorial auctions, Multi-Objective Branch and Bound, Metaheuristics, Hybrid methods, Fuzzy dominance.

1. Introduction

The auctions research started essentially in 1961 with the economist Nobel Prize Laureate William S. Vickrey [37], but the early work on auctions first appeared in operations research journals with Friedman [10] and Rothkopf [29]. Since then, the field of auctions studies has grown to more wide multidisciplinary fields like economics, game theory, operations research, computer science, decision analysis, multicriteria decision making, etc. Numerous applications have been reported in the literature for combinatorial auctions. They have been employed in a variety of industries (truckload transportation, bus routes, industrial procurement) [4], in airport arrival and departure slots [28], in telecommunication (allocating radio spectrum) [5], in electronic business (eBAY) [24], in public sector for procuring meals for schools [17], etc.

In combinatorial auctions, the auctioneer has a set $M = \{a_1, a_2, \dots, a_i, \dots, a_m\}$ of m items to sell, and the bidders submit a set $B = \{b_1, b_2, \dots, b_j, \dots, b_n\}$ of n bids. A bid b_j is a tuple $\langle S_j, c_j\rangle$, where $S_j \subseteq M$ is a combination of a package of items and

c_j is a price for the total package S_j proposed by the j-th bidder. The compelling motivation for the use of combinatorial auctions is the presence of complementarities among the items that differ across bidders (the value of the total package is larger than the sum of the values of its components taken separately). For example:

- A mobile phone operator may value licenses in two adjacent cities more than the sum of the individual licenses values.
- A trucker's cost of handling shipments in one lane depends on its loads in other lanes.

This problem is known as the Winner Determination Problem of Combinatorial Auctions (WDP of CAs) and the selection of the winning bids becomes in this case more complicate (NP-hard problem [30]).

Most studies in the literature focus either on multi-item combinatorial auctions with price only (single-attribute) ([7], [6], [23], [38]) or on single-item (but non-combinatorial auctions) with multi-attribute auctions. However, both auctions types alone are already very complicated and the most research on this area focuses on the computational issues.

The WDP in the Single-unit case consists to label the bids $j = 1, \ldots, n$ as winning ($x_j = 1$) or losing ($x_j = 0$), so as to maximize the auctioneer's revenue under the constraint that each item can be allocated to at most one bidder:

$$(WDP)\left\{\begin{array}{lrll} \max Z(x) = \displaystyle\sum_{j=1}^{n} c_j x_j & & & \\ \text{subject to} & & & \\ & \displaystyle\sum_{j/i\in S_j} x_j & \leq 1 & i = 1, \ldots, m \\ & x_j & \in \{0,1\} & j = 1, \ldots, n \end{array}\right.$$

(WDP) is intractable. The branch and bound algorithms ([31], [32], [33]) are the most common used methods in the single unit case. Exact methods guarantee that an optimal solution is found but do not guarantee the running time! Recently heuristics and approximate methods have been introduced to solve the WDP in combinatorial auctions ([3], [8], [12], [15]).

The availability of multiple units of each item to the WDP involves too many possible combinations to evaluate and so, causes new levels of complications in the auctions process. The auctioneer has some number μ_i : of available units of each item a_i ($i = 1, \ldots, m$). The bidders submit a set of bids $\{b_1, b_2, \ldots, b_j, \ldots, b_n\}$. A bid is a tuple $b_j = \langle\{\lambda_j^1, \lambda_j^2, \ldots, \lambda_j^i, \ldots, \lambda_j^m\}; c_j\rangle$, where λ_j^i is the (non negative integer) number of units of the item a_i ($i = 1, \ldots, m$), required by the j-th bidder ($j = 1, \ldots, n$) and c_j is a price for the total package proposed by the j-th bidder. The corresponding model is given by $(WDP)_\mu$:

$$(WDP)_\mu\left\{\begin{array}{lrll} \max Z(x) = \displaystyle\sum_{j=1}^{n} c_j x_j. & & & \\ \text{subject to} & & & \\ & \displaystyle\sum_{j=1}^{n} \lambda_j^i x_j & \leq \mu_i & i = 1, \ldots, m \\ & x_j & \in \{0,1\} & j = 1, \ldots, n \end{array}\right.$$

Several exact approaches have been used for solving $(WDP)_\mu$: dynamic programming [30], linear programming [26], integer programming [1] and constraint programming [13].

The present work focuses on the allocation problem that has to be solved, in an exact and in an approximate ways, by the auctioneer after all bids are submitted. Furthermore, the submitted bids have to be done taking into account multiple, often conflicting, decision criteria. Thus far, there have not been much work on multi-objective multi-item multi-unit combinatorial auctions and the most of works in this area use the weighted sum approach to translate the multi-objective into an utility objective function ([2], [34]) or use a single objective branch-and-bound algorithm based on the $\varepsilon - constraint$ method [4].

The problem is modeled as a Multi-Objective Multi-Constraint Knapsack Problem (MOMCKP). Indeed, literature search revealed deep connection between these two problems ([14], [16], [17], [27]). We then develop a hybrid MOBB & RWTS approach for the Multi-Objective WDP of CAs (MOWDP of CAs). It combines and exploits the advantages of the Multi-Objective Branch-and-Bound (MOBB) method and the Random Walk Tabu Search (RWTS) metaheuristic. We use here a MOBB method that generalises Ulungu's MOBB [36] to the multi-constraint knapsack problem case with more than two objectives. We compare the performance of our Extended MOBB method to the Florios's et al. MOBB method [9]. The branching sequence has a great impact on the speed convergence of the MOBB approach. We present a novel rule to automatically rank bids while taking into account the DM's preferences on objectives that are most relevant. We then compute a fuzzy dominance relation, on the discrete set of weight vectors, and use it to rank bids and to select a feasible solution (a subset of accepted bids). Seeing that real world MOWDP of CAs instances are not available for solver benchmarking, we validate the performances of the MOBB & RWTS hybrid method on some MOMCKP benchmark instances available in the literature ([18],[19]).

The remainder of this paper is organized as follows. Section 2 presents a multi-objective formulation for a combinatorial auctions problem extending the existing single-objective models and defines the concepts of Pareto dominance and fuzzy dominance that we will use to solve the problem. Section 3 describes the components of the proposed hybrid method, defined by the Extended MOBB and the RWTS metaheuristic. Section 4 presents computational results of both the exact and the hybrid proposed methods and analyzes their performances. Finally, Section 5 concludes with some important obtained results and gives some directions of future works.

2. Multi-Objective Multi-Item Multi-Unit Combinatorial Auction WDPs

We are interested by the problem of CAs in which multiple items are sold and bidders submit bids on packages rather than just individuals items. The seller expresses his preferences upon the suggested complementary items and the buyers are in competition with all the specified attributes done by the seller. Indeed, for each bid, the auctioneer fixes some specified attributes (e.g. maximize the revenue, minimize the payment time, minimize the risk of not being paid after a bid has been accepted, etc). The multi-objective formulation of $(WDP)_\mu$ is:

$$(MOWDP)_\mu \begin{cases} \text{“}opt\text{”}\, Z^k(x) = \sum_{j=1}^{n} c_j^k x_j & k = 1, \ldots, p \\ \text{subject to} & \\ \qquad \sum_{j=1}^{n} \lambda_j^i x_j \leq \mu_i & i = 1, \ldots, m \\ \qquad x_j \in \{0,1\} & j = 1, \ldots, n \end{cases}$$

where

- c_j^k is the value of the bid j $(j = 1, \ldots, n)$ for the criterion k $(k = 1, \ldots, p)$.
- The decision variables are defined as follows:

$$x_j = \begin{cases} 1 & \text{if the bid } b_j \text{ is accepted (the winner offer);} \\ 0 & \text{otherwise.} \end{cases}$$

So, $(MOWDP)_\mu$ consists in finding the accepted bids which, for example, simultaneously maximize the revenue of the seller and minimize the payment time, under the constraints that at most the available number of units of each item is allocated.

An acceptable bid (non risk of overlapping with other bids) for which the vector of specifications (revenue vector) is not dominated by any other vector of specifications of bids, is an *efficient* solution for the $(MOWDP)_\mu$ problem.

2.1. **Pareto Dominance.** The general multi-objective combinatorial optimization problem can be expressed as:

$$(MOCO) \begin{cases} \text{“}\max\text{”} F(x) = \big(f^1(x), f^2(x), \ldots, f^p(x)\big) \\ \text{subject to} \\ \qquad x \in S \end{cases}$$

where p, $(p \geq 2)$ is the number of objective functions, $x = (x_1, x_2, \ldots, x_n)$ is the vector representing the decision variables, S is the (finite) set of feasible solutions in the solution space $\mathbb{R}^n$. The set $Z = F(S)$ represents the feasible points (outcome set) in the objective space $\mathbb{R}^p$ and $z = (z^1, z^2, \ldots, z^p)$, with $z^i = f^i(x)$, for $x \in S$, is a point of the objective space.

Note that in (MOCO), the term “max” appears in quotation marks because, in general, there does not exist a single solution that is maximal on all objectives. As a consequence, several concepts must be established to define what an “optimal” solution is. The more used one is the dominance relation also known as Pareto dominance (see Figure 1).

We recall some basic notions in the theory of multi-objective optimization. For more details, see [35].

Definition 1. *We say that a point $z = (z^1, z^2, \ldots, z^p)$ dominates a point $w = (w^1, w^2, \ldots, w^p)$ and we write $z \succeq w$ if and only if for all $k \in \{1, \ldots, p\}$, $z^k \geq w^k$ with for at least one $l \in \{1, \ldots, p\}$ such that $z^l > w^l$.*

Definition 2. *A solution $x^* \in S$ is called (Pareto) efficient for (MOCO) if and only if there does not exist any other feasible solution $x \in S$, such that $F(x)$ dominates $F(x^*)$. The point $F(x^*)$ is then called a non-dominated point.*

The set of efficient solutions, also called the Pareto optimal set, is often denoted by E and the image of E in Z is called the non-dominated frontier or the Pareto optimal front, and is denoted by Z_E.

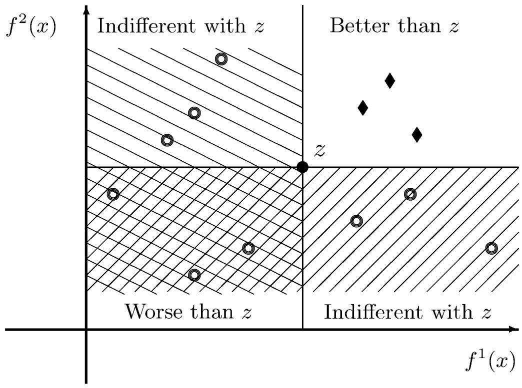

FIGURE 1. Dominations in the Pareto sense in a bi-objective space.

Note that if $x, y \in S$ are such that $F(x)$ dominates $F(y)$ we usually say that x dominates y and we also write $x \succeq y$.

In (MOCO), we can optimize separately each of the objectives by solving the following problems:

$$(COP(k)) \begin{cases} \max \ f^k(x) \qquad k = 1, \ldots, p \\ \text{subject to} \\ \qquad\qquad x \in S \end{cases}$$

Assume that the optimal solutions of the above problems are x^{k*}, $k = 1, \ldots, p$. Then, the optimal value of objective k is given by $f^{k*} = f^k(x^{k*})$.

Definition 3. *The point* $F^* = (f^{1*}, f^{2*}, \ldots, f^{p*})$ *is called the ideal point in the objective space.*

In general, an ideal point is not a feasible solution. Otherwise, the objectives would not be in conflict with one another.

2.2. **Fuzzy Dominance.** The concept of Pareto dominance is refined by introducing the DM's preferences which are the key aspect that must be taken into account in a multi-criteria choice and a ranking process. In this work, we suggest to determinate a fuzzy subset of non-dominated bids according to the DM's preferences on objectives that are most relevant. The fuzzy dominance degree of each bid is then computed and used to rank bids and to select a feasible solution (a subset of accepted bids). Let us consider a set $B = \{b_1, b_2, \ldots, b_n\}$ of bids and a discrete set $\Pi = \{\pi^1, \pi^2, \ldots, \pi^{Ł}\}$ of weight vectors that model the DM's preferences on the objectives. To each bid b_j, we associate the vector $[U(b_j)]^t = (U^1(b_j), U^2(b_j), \ldots, U^{Ł}(b_j))$, representing the multiple utilities of the bid b_j according to the various weight vectors $\pi^1, \pi^2, \ldots, \pi^{Ł}$, with:

$$U^l(b_j) = U(b_j, \pi^l) = \sum_{k=1}^{p} \pi^l_k c^k(b_j), \quad l = 1, \ldots, Ł. \tag{1}$$

To explore the discrete set of weight vectors, we propose to determine the least dominated bid. The credibility of the proposition "b_j is at least as good as b_h" is computed by the following proposed fuzzy dominance relation on $B \times B$:

$$\mu_D(b_j, b_h) = \max(P_U(b_j, b_h) - P_U(b_h, b_j), 0). \tag{2}$$

where $P_U(b_j, b_h)$ represents the proportion of utility for which b_h is not preferred to b_j and is defined by:

$$P_U(b_j, b_h) = \begin{cases} \frac{|\{l \mid U^l(b_j) \geq U^l(b_h)\}|}{\mathrm{L}}, & \text{if } \nexists\, l^0 \mid U^{l^0}(b_j) + v < U^{l^0}(b_h); \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

and v is a threshold of veto (for example, $v = 0.2$).

If for one weight vector l^0, the difference between $U^{l^0}(b_j)$ and $U^{l^0}(b_h)$ is too unfavorable to b_j, then we refuse any credibility to the upgrade of b_h by b_j whatever are the performances of these two bids for the other weight vectors.

For a fixed bid b_j, $D(b_j) = \{b_h \in B : \mu_D(b_j, b_h) > 0\}$ represents the fuzzy subset of bids b_h dominated by b_j. Its complementary, defined by the membership function: $1 - \mu_D(b_j, b_h)$, is the fuzzy subset of bids non-dominated by b_j.

The intersection of all the fuzzy subsets of the bids non-dominated by b_j, when b_j goes through B gives the subset of the bids that are dominated by no other one. The corresponding membership function is defined by:

$$\mu^{ND}(b_h) = \inf\{1 - \mu_D(b_j, b_h), b_j \in B\}. \tag{4}$$

$\mu^{ND}(b_h)$ can be interpreted as the degree of truth of the assertion:

"b_h is dominated by no bids in B".

When we look for the best bids (problem of optimization), it is thus logical to choose the one for which the value of μ^{ND} is closest to 1. If we aim to obtain a complete ranking of bids, it is necessary to proceed by successive steps. This can be done by eliminating the bids already ranked and by recomputing μ^{ND} at every time.

3. Hybridizing Branch & bound and Tabu Search

We develop a hybrid approach (MOBB & RWTS) which combines the MOBB scheme and the RWTS metaheuristic. The MOBB used here generalizes Ulungu's MOBB [36] to the multi-constraint knapsack problem case with more than two objectives. Similarly to the single objective case, the branching sequence has a great impact on the convergence of the algorithm (Martello and Toth [20]; Nemhauser and Wolsey [25]). In this context, a novel branching rule based on the fuzzy dominance relation between bids is proposed, specific for the MOWDP of CAs and can be applied to the general case of $0-1$ Multi-Objective Programming problems. On the other hand, the MOBB method was originally developed for generating the set of Pareto optimal solutions in Mixed Integer Multi-Objective Linear Programming problems of small and medium sizes (Mavrotas and Diakoulaki [21, 22]). It must be noted that the main difference between the mixed integer and the pure integer case is that in the latter, the solution of a Multi-Objective Linear Programming (MOLP) problem in the final nodes is degenerated to simple function evaluations as all binary variables are fixed (there are no continuous variables). Thus, the whole process is significantly faster, as the time consuming part of the generation of the Pareto optimal solutions in each final node's MOLP is avoided. To reduce the running time in solving large NP-hard MOWDP of CAs, we propose to generate the new successor nodes, when the Extended MOBB algorithm progresses deeper down the tree, by the RWTS metaheuristic. Indeed, solving the $(MOWDP)_\mu$ in combinatorial auctions using an exact method may be computationally too long since the number of combinations to be evaluated grows exponentially with the size of the problem!

3.1. **Extended Multi-Objective Branch-and-Bound method.** In the branch-and-bound scheme, the solution space is explored by dynamically building a tree and by using the following three basic procedures: separation, evaluation and sterilization.

Procedure of separation. The branching sequence is crucial for the performance of the method. Let θ be the order according to which variables (bids) of a partial solution will be assigned a value. The order θ can be defined by the decreasing values of $\mu_j^{ND} = \mu^{ND}(b_j),\ \ j = 1, \ldots, n$ (Formula 4) or as in Florios et al. [9] according to the decreasing values of the following heuristics rules:

$$Ave\_sort_j = \frac{1}{p.m} \sum_{k=1}^{p} \sum_{i=1}^{m} \frac{c_j^k}{\lambda_j^i}, \quad j = 1, \ldots, n, \tag{5}$$

$$max_j = \max_{k=1,\ldots,p;\ i=1,\ldots,m} \frac{c_j^k}{\lambda_j^i}, \quad j = 1, \ldots, n, \tag{6}$$

$$S_j = \sum_L X_j^{(L)}, \quad j = 1, \ldots, n, \tag{7}$$

where $X_j^{(L)}$ is the value of the $j-th$ decision variable in the $L-th$ Pareto optimal solution of the relaxed problem. So that larger the sum, more frequent is the presence of the specific variable in the Pareto optimal solution.

Partial solutions (nodes of the search tree) are created by assigning zeros and ones to subsets of bids denoted β_0 and β_1, respectively, and according to the defined order θ. Bids not assigned either zero or one define the set $\mathcal{F} = \{1, 2, \ldots, n\} \setminus \{\beta_0 \cup \beta_1\}$.

For testing the feasibility of the solutions, we define the *matrix of conflicts* between bids (two bids are in conflict if the quantity of a certain item needed by both these two bids is not available). The problem corresponding to the partial solution (β_1, β_0) is again a MOWDP:

$$(MOWDP)_{\overline{\mu}} \begin{cases} \text{“opt”} Z^k(x) = \sum_{j \in \beta_1} c_j^k + \sum_{j \in \mathcal{F}} c_j^k x_j & k = 1, \ldots, p \\ \text{subject to} & \\ \sum_{j \in \mathcal{F}} \lambda_j^i x_j \ \leq \ \overline{\mu}_i & i = 1, \ldots, m \\ x_j \ \in \ \{0, 1\} & j \in \mathcal{F} \end{cases}$$

where:

$$\overline{\mu}_i = \mu_i - \sum_{j \in \beta_1} \lambda_j^i, i = 1, \ldots, m. \tag{8}$$

A solution obtained by assigning a value to all the free variables is called a *completion* of a partial solution.

Procedure of evaluation. An evaluation function is a vector that is simultaneously fast to calculate and close to the Pareto front. We can evaluate the subset S' of S if we know how to determine a vector $g(S') = (g^1(S'), \dots, g^p(S'))$ in such a way that there exists no solution $\overline{s} \in S'$ such that $Z(\overline{s})$ *dominates* $g(S')$. If $S' = \emptyset$, then the only possible evaluation is $g(S') = -\infty$ for a maximization problem ($g(S') = +\infty$ for a minimization problem).

For each node S' of the tree, we associate a vector valued bounds. To compute the bounds, we define for $k = 1, \dots, p$:

- $\sum\limits_{j \in \beta_1} c_j^k$: the value of the bids that have already been assigned value 1.
- $Z^{k*}(S')$: the value of the optimal solution according to the k-th objective.

The values of $\sum\limits_{j \in \beta_1} c_j^k$ and of $\Big(Z^{k*}(S') + \sum\limits_{j \in \beta_1} c_j^k\Big)$ $(k = 1, \dots, p)$ represent both an estimation and an evaluation of the subset S' respectively. If we consider two objectives, the first is to maximize and the second is to minimize, the lower bound $\underline{Z}(S')$ and the upper bound $\overline{Z}(S')$ at the partial solution S', are given by:

$$\underline{Z}(S') = \underline{z} = (\underline{z}_1, \underline{z}_2) = \Big(\sum_{j \in \beta_1} c_j^1, \big(Z^{2*}(S') + \sum_{j \in \beta_1} c_j^2\big)\Big); \tag{9}$$

$$\overline{Z}(S') = \overline{z} = (\overline{z}_1, \overline{z}_2) = \Big(\big(Z^{1*}(S') + \sum_{j \in \beta_1} c_j^1\big), \sum_{j \in \beta_1} c_j^2\Big). \tag{10}$$

The components of both the lower bound $\underline{Z}(S')$ and the upper bound $\overline{Z}(S')$ at the partial solution S', are respectively given according to Table 1.

Table 1. Lower and upper bounds at a partial solution

Bounds	maximization problem	minimization problem
Lower bound	estimation	evaluation
Upper bound	evaluation	estimation

Procedure of sterilization. A subset S' of the set S of solutions of a multi-objective combinatorial auctions problem is said to be pruned if $S' = \emptyset$ or there exists $s^* \in S$ so that s^* dominates any solution of S'.

The vector valued bounds, defined above, allows us to prune a partial solution $S^{'}$, when no *completion* of S' can possibly contain an efficient solution. So, we do not need to develop further the exploration of such a node.

Extended MOBB Algorithm. Without loss of generality, assume that the bids b_1, b_2,..., b_n are sorted in such a way that:

$$\mu^{ND}(b_1) \geq \mu^{ND}(b_2) \geq \dots \geq \mu^{ND}(b_n). \tag{11}$$

The first bid to choose and to include in the selection is the one with the largest μ^{ND} value. Then, the bid having the second largest μ^{ND} in the collection is accepted if its acceptance does not create any conflict with bids already selected, otherwise it is rejected. And so on, until all bids are reviewed. We get a subset of bids that may be a feasible solution to $(MOWDP)_\mu$. Let $l_* = \min\{l : \mu \not\geq \lambda^l\}$, with

$$(\lambda^l)^t = \left(\sum_{j=1}^{l} \lambda_1^j, \sum_{j=1}^{l} \lambda_2^j, \ldots, \sum_{j=1}^{l} \lambda_m^j\right)$$

and

$$\mu^t = (\mu_1, \mu_2, \ldots, \mu_m).$$

Thus, l_* is the smallest index l such that

$$\exists\, i_0 \mid \mu_{i_0} < \sum_{j=1}^{l} \lambda_{i_0}^j.$$

Note that the critical bid is b_{l_*}. So, the subset $S' = \{b_1, b_2, \ldots, b_{l_*-1}\}$ of bids is a feasible solution because all bids in S' are not in conflict i.e. they have not items in common. The Extended MOBB method starts by fixing many bids according to the θ order, defined by the proposed fuzzy rule (Formula 4), to quickly find a good feasible solution. Thus, many branches of the tree can be pruned early.

The list $\mathcal{N}$ of nodes is maintained as a $LIFO$ stack (Last In First Out). When a node is pruned, the algorithm backtracks and creates a new node by moving the last bid in β_1 to β_0. In addition, all bids in β_0 after this new bid become free. If, however, n was the last bid in β_1, the algorithm removes all bids $\{v, \ldots, n\}$ in β_1 (v is the smallest one) and defines β_0 to be all previous elements of β_0 up to $v-1$ and to include v. When a node is not pruned, the algorithm progresses deeper down the tree and creates a new successor node. Indeed, as many bids as possible are included in β_1, according to order θ, i.e. as they appear in $\mathcal{F}$. But if the remaining vector $\overline{\mu}$ does not allow bid l to be added to β_1, the first possible bid r of $\mathcal{F}$, which can be added to β_1 is sought and bid r is added to β_1. Of course, all bids $\{i, \ldots, r-1\}$ must be added to β_0. The method is summarized in Algorithm 1.

Algorithm 1 Extended MOBB for $(MOWDP)_\mu$.

Require: Data of $(MOWDP)_\mu$.
Ensure: A set of efficient solutions.
  Initialization: Create the root node N_0 as follows:

$$\beta_1 := \emptyset,\ \beta_0 := \emptyset,\ \mathcal{F} := \{1, \ldots, n\},\ \mathcal{L} := \phi,\ \mathcal{N} := \{N_0\}.$$

  while $\mathcal{N} \neq \emptyset$ **do**
    Choose the last node $N \in \mathcal{N}$.
    Compute $\underline{z}$ (lower bound at the partial solution).
    Add $\underline{z}$ to $\mathcal{L}$ if it is not dominated.
    Compute $\overline{z}$ (an upper bound at the partial solution).
    if $\{j \in \mathcal{F} : bid$ j is not in conflict with $\beta_1\} = \phi$ **OR** $\overline{z}$ is dominated by some $y \in \mathcal{L}$ **then**
      (Case 1.)
      Prune the node N. $\mathcal{N} := \mathcal{N} \setminus \{N\}$.
      Go backwards of the node N (node N is pruned).
      Create a new node N'.
      Update β_1, β_0 and $\mathcal{F}$.
      $\mathcal{N} := \mathcal{N} \cup \{N'\}$.
      if the set β_1 of N' is smaller than β_1 of the predecessor nodes of N, which are not predecessors of N', **then**
        Fathom these nodes.
      end if
      if $(\beta_1 = \phi)$ **then**
        Compute the upper bound, $\overline{z}$, at this partial solution.
        if there is a solution in $\mathcal{L}$ which dominates $\overline{z}$ **then**
          STOP (no new node can be created).
        end if
      end if
    else
      (Case 2.)
      Go deeper down the tree (node N is not pruned).
      Create a new node N'.
      Update β_1, β_0 and $\mathcal{F}$.
      $\mathcal{N} := \mathcal{N} \cup \{N'\}$.
    end if
  end while

Didactic Example. The Extended MOBB method presented in Algorithm 1 is illustrated by means of a multi-objective combinatorial auctions problem with three objective functions, three constraints and seven binary variables. Let be:

- $M = \{a_1, a_2, a_3\}$ the set of three items to be auctioned.
- $\mu_1 = 5$, $\mu_2 = 10$ and $\mu_3 = 7$ (number of available units).

The offers b_j, $j = 1, \ldots, 7$ upon the set M and their revenue 3-vectors c_j (to be maximized) are given as follows:

- $b_1 = \langle\{1, 2, 3\}; c_1 = (10, 5, 12)\rangle$. This means that the bid b_1 contains one unit of item a_1, two of a_2 and three of a_3 and c_1 is its revenue vector.

- $b_2 = \langle\{1,3,2\}; c_2 = (6,10,8)\rangle$
- $b_3 = \langle\{4,6,4\}; c_3 = (7,14,5)\rangle$
- $b_4 = \langle\{1,3,0\}; c_4 = (9,17,4)\rangle$
- $b_5 = \langle\{5,2,0\}; c_5 = (6,9,3)\rangle$
- $b_6 = \langle\{1,4,0\}; c_6 = (13,6,11)\rangle$
- $b_7 = \langle\{2,7,1\}; c_4 = (5,16,4)\rangle$

The conflict graph is given in Figure 2.

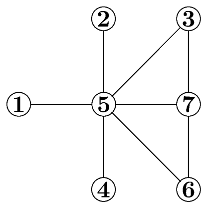

FIGURE 2. The conflict graph of the example: a vertex i is adjacent to a vertex j if the bids i and j can not be chosen together

The problem can be formulated by the following mathematical program:

$$\left\{\begin{array}{lllllllll}
\max Z^1(x) = 10x_1 + & 6x_2 + & 7x_3 + & 9x_4 + & 6x_5 + & 13x_6 + & 5x_7 & & \\
\max Z^2(x) = \;\; 5x_1 + & 10x_2 + & 14x_3 + & 17x_4 + & 9x_5 + & 6x_6 + & 16x_7 & & \\
\max Z^3(x) = 12x_1 + & 8x_2 + & 5x_3 + & 4x_4 + & 3x_5 + & 11x_6 + & 4x_7 & & \\
\text{subject to} & & & & & & & & \\
\qquad x_1 + & x_2 + & 4x_3 + & x_4 + & 5x_5 + & x_6 + & 2x_7 & \leq & 5 \\
\qquad 2x_1 + & 3x_2 + & 6x_3 + & 3x_4 + & 2x_5 + & 4x_6 + & x_7 & \leq & 10 \\
\qquad 3x_1 + & 2x_2 + & 4x_3 + & & & & x_7 & \leq & 7 \\
\qquad x_1, & x_2, & x_3, & x_4, & x_5, & x_6, & x_7 & \in & \{0,1\}
\end{array}\right.$$

The order $\theta = \{x_4, x_6, x_1, x_7, x_3, x_5, x_2\}$ is computed according to the fuzzy first rule (Formula 5).

- The node N_0 is created, $\beta_1 := \emptyset, \beta_0 := \emptyset, \mathcal{F} := \{4,6,1,7,3,5,2\}$, $\underline{z} := \begin{pmatrix} 0 \\ 0 \end{pmatrix}$, $\overline{z} := \begin{pmatrix} \infty \\ \infty \end{pmatrix}$, $\mathcal{L} := \emptyset$, $\mathcal{N} := \{N_0\}$.
- The node N_1 is created, $\beta_1 := \{4,6,1\}, \mathcal{F} := \{7,3,5,2\}$, $\overline{\mu}^t = (2,1,4)$, $\underline{z}^t := (32,28,27)$, $\mathcal{L} := \{(32,28,27)^t\}$, $\{j \in \mathcal{F} | \lambda_j \leq \overline{\mu}\}$=$\emptyset$ then the node N_1 is pruned.
- The node N_2 is created, $\beta_1 := \{4,6\}, \beta_0 := \{1\}, \mathcal{F} := \{7,3,5,2\}$, $\overline{\mu}^t = (3,3,7)$, $\underline{z}^t := (22,23,15)$, the computation of the $\overline{z}$ does not allow to prune this node.
- The node N_3 is created, $\beta_1 := \{4,6,2\}, \beta_0 := \{1,7,3,5\}, \mathcal{F} := \emptyset$, $\overline{\mu}^t = (2,0,5)$, $\underline{z}^t := (28,33,23) = \overline{z}$, $\mathcal{L} := \{(32,28,27)^t, (28,33,23)^t\}$, $\{j \in \mathcal{F} | \lambda_j \leq \overline{\mu}\}$=$\emptyset$, then the node N_3 is pruned.
- The node N_4 is created, $\beta_1 := \{4\}, \beta_0 := \{6\}, \mathcal{F} := \{1,7,3,5,2\}$, $\overline{\mu}^t = (4,7,7)$, $\underline{z}^t := (9,17,4)$. Thus, N_2 is pruned because β_1 of N_4 is smaller

than β_1 of the predecessor of N_3, which is N_2, and the computation of $\overline{z}$ does not allow to prune this node.

- The node N_5 is created, $\beta_1 := \{4,1\}, \beta_0 := \{6\}, \mathcal{F} := \{7,3,5,2\}$, $\overline{\mu}^t = (3,5,4)$, $\underline{z}^t := (19,22,16)$, the computation of $\overline{z}$ does not allow to prune this node.
- The node N_6 is created, $\beta_1 := \{4,1,2\}, \beta_0 := \{6,7,3,5\}, \mathcal{F} := \emptyset$, $\overline{\mu}^t = (2,2,2)$, $\underline{z}^t := (25,32,24) = \overline{z}^t$.
$\mathcal{L} := \{(32,28,27)^t, (28,33,23)^t, (25,32,24)^t\}$, $\{j \in \mathcal{F} | \lambda_j \leq \overline{\mu}\}$=$\emptyset$,
then the node N_6 is pruned.
- The node N_7 is created, $\beta_1 := \{4\}, \beta_0 := \{6,1\}, \mathcal{F} := \{7,3,5,2\}$, $\overline{\mu}^t = (4,7,7)$, $\underline{z}^t := (9,17,4)$, N_5 is pruned since β_1 de N_7 is smaller than β_1 of N_5, the predecessor of N_6. The computation of $\overline{z}$ does not allow to prune this node.
- The node N_8 is created, $\beta_1 := \{4,7\}, \beta_0 := \{6,1\}, \mathcal{F} := \{3,5,2\}$, $\overline{\mu}^t = (2,0,6)$, $\underline{z}^t := (14,33,8) = \overline{z}^t$, $\{j \in \mathcal{F} : \lambda_j \leq \overline{\mu}\}$=$\emptyset$. Then the node N_8 is pruned.
- The node N_9 is created, $\beta_1 := \{4\}, \beta_0 := \{6,1,7\}, \mathcal{F} := \{3,5,2\}$, $\overline{\mu}^t = (4,7,7)$, $\underline{z}^t := (9,17,4)$, the computation of $\overline{z}$ does not allow to prune this node.
- The node N_{10} is created, $\beta_1 := \{4,3\}, \beta_0 := \{6,1,7\}, \mathcal{F} := \{5,2\}$, $\overline{\mu} = (0,1,3)$, $\underline{z}^t := (16,31,9) = \overline{z}^t$, $\{j \in \mathcal{F} : \lambda_j \leq \overline{\mu}\}$=$\emptyset$. Then the node N_{10} is pruned.
- The node N_{11} is created, $\beta_1 := \{4\}, \beta_0 := \{6,1,7,3\}, \mathcal{F} := \{5,2\}$, $\overline{\mu} = (4,7,7)$, $\underline{z}^t := (917,4)$, $\overline{z}^t := (18,32,13)$,and is then dominated by bids in $\mathcal{L}$ then the node N_{11} is pruned.
- The node N_{12} is created, $\beta_1 := \emptyset, \beta_0 := \{4\}, \mathcal{F} := \{6,1,7,3,5,2\}$, $\overline{\mu}^t = (5,10,7)$, $\underline{z}^t := (0,0,0)$, the nodes N_4, N_7, N_9 are pruned since β_1 de N_{12} is smaller than β_1 of predecessor of N_{11}.
the computation of $\overline{z}$ does not allow to prune this node.
- The node N_{13} is created,$\beta_1 := \{6,1\}, \beta_0 := \{4\}, \mathcal{F} := \{7,3,5,2\}$, $\overline{\mu}^t = (3,4,4)$, $\underline{z}^t := (23,11,23) = \overline{z}^t$, the computation of $\overline{z}$ does not allow to prune this node.
- The node N_{14} is created, $\beta_1 := \{6,1,2\}, \beta_0 := \{4,7,3,5\}, \mathcal{F} := \emptyset$, $\overline{\mu}^t = (2,1,2)$, $\underline{z}^t := (29,21,31) = \overline{z}^t$,
$\mathcal{L} := \{(32,28,27)^t, (28,33,23)^t, (25,32,24)^t, (29,21,31)^t\}$
$\{j \in \mathcal{F} : \lambda_j \leq \overline{\mu}\}$=$\emptyset$ then the node N_{14} is pruned.
- The node N_{15} is created, $\beta_1 := \{6\}, \beta_0 := \{4,1\}, \mathcal{F} := \{7,5,3,2\}$, $\overline{\mu}^t = (4,6,7)$, $\underline{z}^t = (13,6,11)$, $\overline{z}^t = (23,25,21)$, the node N_{13} is pruned because β_1 of N_{15} is smaller than the predecessor of N_{14} which is N_{13}.
$\overline{Z}$ is dominated by bids in $\mathcal{L}$ then the node N_{15} is pruned.
- The node N_{16} is created, $\beta_1 := \emptyset, \beta_0 := \{4,6\}, \mathcal{F} := \{1,7,3,5,2\}$, $\overline{\mu}^t = (5,10,7)$, $\underline{z}^t = (0,0,0)$, $\overline{z}^t = (21,28,23)$, $\overline{z}$ is dominated by bids in $\mathcal{L}$ then the node N_{16} is pruned. We can not create an other node. Then N_0 is pruned and the algorithm stops.

The solution process of the Didactic Example is illustrated by Figure 3.

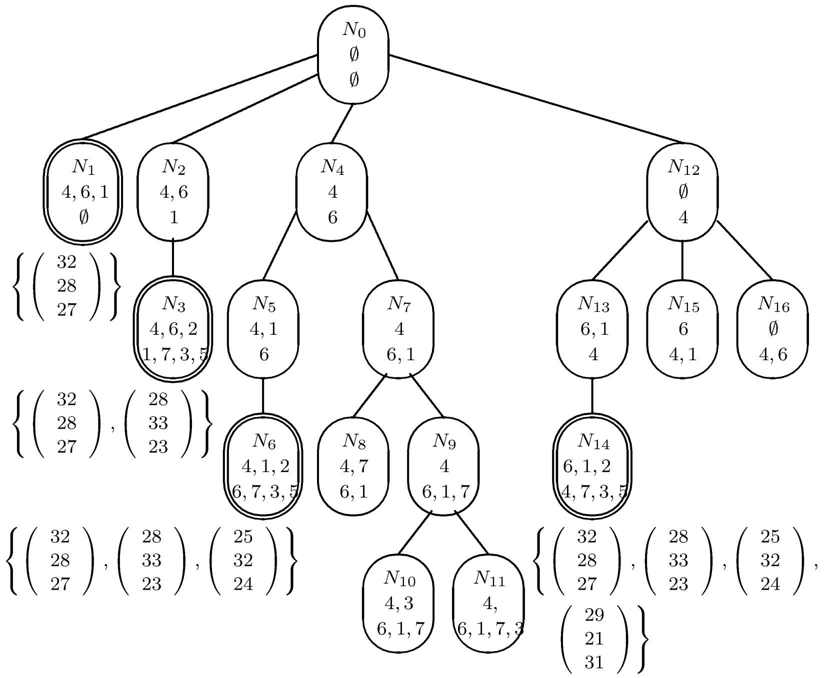

FIGURE 3. Tree of the branch-and-bound search of the Didactic Example

The nodes are indexed in the order according to which they are created, and are represented by the sets β_1, β_0 and $\mathcal{L}$. There are four efficient combinatorial auctions, summarized in Table 2.

TABLE 2. Solutions of the Didactic Example

N°	**Efficient combinatorial auctions**	**Revenue vector**
1	$\{b_4, b_6, b_1\}$	(32,28,27)
2	$\{b_4, b_6, b_2\}$	(28,33,23)
3	$\{b_4, b_1, b_2\}$	(25,32,24)
4	$\{b_6, b_1, b_2\}$	(29,21,31)

3.2. **Tabu Search method.** In the resolution of multi-objective combinatorial optimization problems by metaheuristic methods, we are always brought to compare solutions. Every new solution is compared with all the previous computed solutions in the temporary set of potentially efficient solutions. As we progress in the resolution, the cardinal of this set increases. This slows down the process of resolution. Tabu Search (TS) is a local search strategy [11] used for intensifying the research and well designed for escaping from local minima. We propose one adaptation of the TS method to $(MOWDP)_\mu$ using a fuzzy dominance relation in the process of comparison between solutions. The main components of a TS procedure are:

Research space. One feasible solution is given by a binary vector $x = (x_1, x_2, \ldots, x_n)$ which verifies all constraints i.e.,

$$\forall i \in \{1, \ldots, m\}, \sum_{j=1}^{n} \lambda_j^i x_j \leq \mu_i.$$

Research space S is then composed of all these binary vectors, i.e.:

$$S = \left\{ x \in \{0,1\}^n \mid \sum_{j=1}^{n} \lambda_j^i x_j \leq \mu_i, i \in \{1, \ldots, m\} \right\}.$$

An initial solution is generated at the same way as in the Extended MOBB Algorithm (see Subsection 3.1.).

Neighborhood. The neighborhood $\mathcal{N}$ of our problem is defined in the following way: let x and x' in S, x and x' are neighbors if and only if they differ exactly in one component. It results that from a current solution x, it is possible to obtain a neighbor solution x' ($x' \in \mathcal{N}(x)$) by adding or removing a bid such that x' remains feasible (it does not conflict with all previous accepted bids). The movement from x to x' is then characterized by the integer j which is considered to be the attribute of movement and represents the index of the component x_j that was changed i.e., ($x_j : 0 \to 1$ or $1 \to 0$).

Evaluation of the Neighborhood. The evaluation between two neighbors $x = (x_1, x_2, \ldots, x_n) \in S$ and $x' = (x'_1, x'_2, \ldots, x'_n) \in \mathcal{N}(x)$, is based on a fuzzy dominance relation. The corresponding membership function, $\mu_D(x', x)$ (Formula 2), measures the credibility of the proposition

"x' is at least as good as x".

Management of the tabu list. Every time a movement is applied to go from the current solution x, to the neighbor solution x'. To avoid cycling cases in which we would come back to x and oscillate between x' and x, the indication of the attribute of the movement is registered in a tabu list. So, the inverse movement (which corresponds on the way back to the departure configuration) is forbidden for a certain number of subsequent moves. Note ND the fuzzy set of the temporary potential non-dominated solutions. For any $y \in ND$, we compute $\mu_D(x', y)$. $ND \leftarrow ND \setminus \{y \in ND : \mu_D(x', y) > 0\}$ (we remove from ND the fuzzy subset of solutions dominated by x').

RWTS algorithm. Contrary to the basic tabu search algorithm, where the diversification is ensured only by the tabu list, RWTS algorithm consists of realizing from time to time a move which is no more guided by the evaluation function and then constitutes a diversification diagram. At every iteration of the RWTS algorithm, a real value $rw \in [0, 1]$ is randomly generated. Let us put $q \in [0, 1]$ the value threshold, then, if $rw > q$ the algorithm will select the best movement, otherwise, the algorithm will make a feasible random movement. The RWTS algorithm can be so described by Algorithm 2.

Algorithm 2 RWTS algorithm with a fuzzy dominance relation

```
Require: A number of iterations nb − iter,  A random threshold q.
Ensure: A fuzzy set ND of potentially non-dominated solutions.
for i = 0 to nb − iter do
    Generate a random value rw ∈ [0, 1].
    if rw ≤ q then
        choose a random allowed move j*.
    else
        choose the best allowed move j*.
    end if
    Update the tabu list with j*.
    Perform the chosen move j* in x: let x′ be the obtained solution.
    Update the fuzzy set of non-dominated solutions with x′.
end for
```

A hybrid approach, presented in Algorithm 3, combines the MOBB method provided in Algorithm 1 and a RWTS metaheuristic presented in Algorithm 2. In our proposed hybrid approach, the previous described Extended MOBB method is referred to be the process of the principal research. When a node of the spanning tree is not pruned and its neighborhood is explored, we suggest to use the RWTS metaheuristic in order to reduce the size of the search space.

Algorithm 3 Hybrid MOBB & RWTS algorithm.

Require: Data of $(MOWDP)_\mu$ problem.
Ensure: A set of potentially efficient solutions.
  Initialization: Create the root node N_0 as follows:

$$\beta_1 := \emptyset,\ \beta_0 := \emptyset,\ \mathcal{F} := \{1, \dots, n\},\ \mathcal{L} := \phi,\ \mathcal{N} := \{N_0\}.$$

  while $\mathcal{N} \neq \emptyset$ **do**
    Choose the last node $N \in \mathcal{N}$.
    Compute $\underline{z}$ (lower bound at the partial solution).
    Add $\underline{z}$ to $\mathcal{L}$ if it is not dominated.
    Compute $\overline{z}$ (an upper bound at the partial solution).
    if $\{j \in \mathcal{F}$: b_j is not in conflict with $\beta_1\} = \emptyset$ **Or** $\overline{z}$ is dominated by some $w \in \mathcal{L}$ **then**
      (Case 1.)
      Prune the node N. $\mathcal{N} := \mathcal{N} \setminus \{N\}$.
      Go backwards of the node N (node N is pruned).
      Create a new node N'.
      Update β_1, β_0 and $\mathcal{F}$.
      $\mathcal{N} := \mathcal{N} \cup \{N'\}$.
      if the set β_1 of N' is smaller than β_1 of the predecessor nodes of N, which are not predecessors of N', **then**
        Fathom these nodes.
      end if
      if $\beta_1 = \emptyset$ **then**
        Compute the upper bound, $\overline{z}$, at this partial solution.
        if there is a solution in $\mathcal{L}$ which dominates $\overline{z}$ **then**
          STOP (no new node can be created).
        end if
      end if
    else
      (Case 2.)
      Go deeper down the tree (node N is not pruned).
      Create a new node N' by RWTS method.
      Update β_1, β_0 and $\mathcal{F}$.
      $\mathcal{N} := \mathcal{N} \cup \{N'\}$.
    end if
  end while

4. Experimental results

Real world MOWDP of CAs instances are not available for solver benchmarking. As literature search revealed deep connection between the WDP of CAs and Knapsack Problems, the performances of the proposed methods were validated on a set of MOMCKP instances.

We focus our experiments on a three objectives ($p = 3$), three items ($m = 3$) and $n = 10, 20, 30, 40, 50$ bids. However, the results remain valid for a larger number of objectives. A tabu list length is fixed to 15 for all the instances and the threshold value for random walk q is set to $0,15$. Except for the exact algorithm, each instance is re-run ten times with the hybrid (MOBB & RWTS) algorithm, and each run terminates after $nb-iter = 1000$ iterations. The algorithms have been implemented in C++ language, using a Pentium PC with dual core processor, FSB 800 Mb, DDR1 2 Go in Windows operating system.

The two MOBB methods are executed on the same machine. The Extended MOBB is compared, on all branching heuristics (Formulas 5-7), to the Florios's et al. MOBB [9] according to five Evaluation criteria: Generated nodes, Dominated nodes, Non Dominated nodes, set of Efficient solutions (E) and CPU Time (seconds).
The experimental results are provided in Tables 3-12 respectively.

TABLE 3. 3kp10: Florios's et al. MOBB [9].

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
No-SORT	614	117	31	9	0.05
AVG-SORT	390	77	13	9	0.04
MAX-SORT	702	170	38	9	0.06
Relax-Sum-Round	376	76	10	9	0.04

TABLE 4. 3kp10: Extended MOBB.

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
NO-SORT	184	75	13	9	0.01173
AVG-SORT	145	57	9	9	0.01043
MAX-SORT	227	91	12	9	0.01029
Relax-Sum-Round	147	58	9	9	0.00863
Fuzzy-RAND	137	52	9	9	**0.00844**

The experiments of the Tables 3, 4 are performed on an Intel machine Pentium (R) CPU p6200 @2.13 GHz with 2 GO RAM and OS is Microsoft Windows 7 Professional in order to be able to compare the two approaches since the execution time on a more powerful machine is exactly equal to zero seconds.

TABLE 5. 3kp20: Florios's et al. MOBB [9].

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
No-SORT	8262	2710	91	61	1.82
AVG-SORT	7032	2497	78	61	0.82
MAX-SORT	8262	2710	91	61	0.88
Relax-Sum-Round	10494	3488	58	61	1.21

Table 6. 3kp20: Extended MOBB.

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
NO-SORT	11052	6987	140	61	0.43
AVG-SORT	5018	3239	72	61	0.19
MAX-SORT	5301	3465	71	61	0.21
Relax-Sum-Round	6827	4249	65	61	0.26
Fuzzy-RAND	3481	2133	88	61	**0.15**

Table 7. 3kp30: Florios's et al. MOBB [9].

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
No-SORT	518170	214006	1190	195	83.16
AVG-SORT	518170	214006	1190	195	83.92
MAX-SORT	317424	115670	482	195	56.46
Relax-Sum-Round	252140	105738	237	195	49.38

Table 8. 3kp30: Extended MOBB.

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
NO-SORT	491647	366335	1129	195	25.73
AVG-SORT	291526	207965	417	195	15.21
MAX-SORT	301271	223766	757	195	16.49
Relax-Sum-Round	224478	167537	255	195	13.45
Fuzzy-RAND	195203	139362	473	195	**10.94**

Table 9. 3kp40: Florios's et al. MOBB [9].

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
No-SORT	6052616	2860391	4488	389	1813.31
AVG-SORT	4260944	2002044	1385	389	1197.92
MAX-SORT	3038572	1429995	2462	389	815.09
Relax-Sum-Round	1781504	812882	560	389	522.83

Table 10. 3kp40: Extended MOBB.

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU (s)
NO-SORT	7640324	6236885	3937	389	750.11
AVG-SORT	4069419	3142036	1372	389	275.00
MAX-SORT	3896463	3246504	2387	389	276.90
Relax-Sum-Round	1992094	1590024	598	389	149.90
Fuzzy-RAND	1811364	1465302	1183	389	**146.16**

For the instance 3kp50 (Table 11 and Table 12), we do not have the order of consideration of the bids according to the Relax-Sum-Round heuristic. This can not influence the performance of the Florios's et al. MOBB because the MAX-SORT heuristic seems to be the best one for this approach.

Table 11. 3kp50: Florios's et al. MOBB [9].

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
No-SORT	458980078	229175911	16881	1048	> 432000.23
AVG-SORT	141600928	70315836	2811	1048	119796.57
MAX-SORT	78531540	38940050	4178	1048	76910.68
Relax-Sum-Round	-	-	-	-	-

Table 12. 3kp50: Extended MOBB.

Branching heuristics	Evaluation criteria				
	Generated nodes	Dominated nodes	Non Dominated nodes	Efficient solutions	CPU(s)
NO-SORT	798660547	706934379	16118	1048	166564
AVG-SORT	411712286	307918571	2802	1048	47866.60
MAX-SORT	379178309	288296690	4146	1048	43598
Relax-Sum-Round	-	-	-	-	-
Fuzzy-RAND	63779291	52990362	3017	1048	**10077.90**

Compared to the Florios's et al. MOBB method, on the branching heuristics (Formulas 5-7), the obtained results from our Extended MOBB method outperform those obtained by Florios's et al. MOBB method in Ratio of Dominated nodes and CPU time evaluation criteria.

Table 13. Results of the Ratio of Dominated nodes (%).

Instances	Ratio of Dominated nodes (%)					
	Florios's et al. MOBB			Extended MOBB		
	min	mean	max	min	mean	max
3kp10	19.05	19.55	24.22	37.96 (+18.91)	38.93 (+19.38)	40.76 (+16.54)
3kp20	32.80	33.59	35.51	61.27 (+24.47)	62.55 (+28.96)	65.36 (+29.85)
3kp30	36.44	40.24	41.94	71.34 (+34.90)	72.54 (+32.30)	74.63 (+32.69)
3kp40	45.63	46.74	47.26	77.21 (+31.58)	80.69 (+33.95)	83.32 (+36.06)
3kp50	49.58	49.72	49.93	74.79 (+25.21)	80.60 (+30.88)	88.51 (+38.58)

The closer to 100 is the Ratio of Dominated nodes the better is the speed of convergence of the MOBB method. Our Extended MOBB method improves, on average, the Ratio of Dominated nodes between +19.38% and +33.95% (see Table 13) which allows to significantly reduce the size of the tree since all these nodes will be sterilized.

Table 14. Results of the CPU Time (seconds).

Instances	CPU Time (s)						
	Florios's et al. MOBB			Extended MOBB			
	min	mean	max	min	mean	max	Fuzzy-Rand
3kp10	0.04	0.038	0.06	0.00863	0.01027	0.01173	0.00844 (-78%)
3kp20	0.82	1.18	1.82	0.19	0.27	0.43	0.15 (-87%)
3kp30	49.38	68.23	83.92	13.45	17.72	25.73	10.94 (-84%)
3kp40	522.83	1087.29	1813.31	149.90	362.98	750.11	176.16 (-84%)
3kp50	76910.68	> 209569	> 432000	43598	86009.53	166564	10077.90 (-95%)

As it is observed in Table 14, our Extended MOBB seems to be comparable, to the Florios's et al. MOBB, for the small sizes and becomes clearly more faster than Florios's et al. MOBB for large sizes. Furthermore, the use of the proposed fuzzy branching heuristic accelerates significantly the MOBB convergence. Indeed, compared to the mean time consumed in seconds by Florios's et al. MOBB our Extended MOBB method with fuzzy branching heuristic improves the CPU Time between −78% and −95%.

The complete set of efficient bids can be generated in reasonable computational time only for small problems. For large problems, it seems better to use approximate methods. The performance of the proposed hybrid method is evaluated in terms of proportion of efficient solutions, generated by the set of Non Dominated solutions (ND), and computational time represented by CPU Time and measured in seconds. The numerical results are reported in Table 15.

Table 15. Performance of Hybrid MOBB & RWTS compared to Extended MOBB with Fuzzy-Rand heuristic.

Instances	Evaluation Criteria					
	$\lvert E\rvert$	$\frac{\lvert E\cap ND\rvert}{\lvert E\rvert}$	CPU Time(s)			
	Extended MOBB	Hybrid MOBB & RWTS	Extended MOBB	Hybrid MOBB & RWTS		
				min	mean	max
3KP10	9	1.00	0.00844	0.00185	0.00187	0.00254
3KP20	61	0.88	0.15	0.01699	0.02430	0.03791
3KP30	195	0.84	10.94	1.02998	1.39047	1.86323
3KP40	389	0.71	176.16	3.75808	8.45568	15.81212
3KP50	1048	0.64	10077.90	66.32464	155.19966	310.39932

5. Conclusion and perspectives

The proposed hybrid method (MOBB & RWTS) for multi-item multi-unit WDP of CAs in the context of multi-attribute (multi-objective) is based on the MOBB approach and RWTS metaheuristic. The MOBB used here is referred to be the process of the principal research. Literature search revealed deep connection between the WDP of CAs and Knapsack Problems. We have extended the Ulungu's MOBB method to the multi-constraint case with more than two objectives, in order to make it possible of handling more than one item (multi-item). A didactic example was presented to illustrate this generalization. The extended MOBB method was compared to Florios's et al. MOBB on some branching heuristics. The results show that our Extended MOBB outperforms Florios's et al MOBB and confirm that the branching sequence has a great impact on the convergence of the MOBB approach. A novel branching rule based on the fuzzy dominance relation between bids was proposed. It accelerates significantly the MOBB convergence. In addition, this improvement is more effective when the size of the problem becomes more large. The complete set of efficient bids can be generated in reasonable computational time only for small problems. For large problems, it seems better to use approximate methods. One adaptation of the RWTS metaheuristic using a fuzzy dominance relation in the process of comparison between solutions was proposed for $(MOWDP)_\mu$ and incorporated into the Extended MOBB. To solve large scale NP-hard combinatorial auctions, the use of hybrid methods seems to be the most promising approach to $(MOWDP)_\mu$ problems. These methods generate good approximated solutions in a short computational time. In future works, we propose to develop other mechanisms of diversification and intensification more successful than the RWTS and to incorporate them into the multi-objective branch-and-bound. Also, the speed of the algorithm can be improved by developing better bounds for the studied problem to discard partial solution that cannot lead to new non dominated criterion vectors. More experiments are needed for large sizes and relations with other combinatorial problems (bin packing, etc) will be examined.

References

[1] A. Anderson, M. Tenhunen, and F. Ygge, *Integer programming for combinatorial auction winner determination*, Proceedings of 4th International Conference on MultiAgent Systems, IEEE Computer Society Press, July (2000), 39-46.

[2] M. Bichler, *An experimental analysis of multi-attribute auctions*, Decision Support Systems, 29 (2000), 249-268.

[3] T. Buer, and H. Kopfer, *A Pareto-metaheuristic for a bi-objective winner determination problem in a combinatorial reverse auction*, Computer & Operation Research, 41 (2014), 208-220.

[4] T. Buer, and G. Pankratz, *A Bi-Objective Winner-Determination Problem in a Transportation-Procurement Auction*, Working paper No 488, Faculty of Business Administration and Economics, University of Hagen, Germany (2010).

[5] P. Cramton, *Spectrum Auctions*, Handbook of Telecommunications Economics, Elsevier Science, Chapter 14, (2002), 605-639.

[6] S. De Vries, and R. Vohray, *Combinatorial auction: A survey*, Tech. rep., Department of Managerial Economics and Decision Sciences, Kellogg School of Management, Northwestern University (2000).

[7] M.-B. Dowlatshahi, and V. Derhami, *Winner Determination in Combinatorial Auctions using Hybrid Ant Colony Optimization and Multi-Neighborhood Local Search*, Journal of AI and Data Mining Vol 5, No 2, (2017), 169-181.

[8] S. Farzi, *Discrete Quantum-Behaved Particle Swarm Optimization for the Multi-Unit Combinatorial Auction Winner Determination Problem*, Journal of Applied Sciences, IISSN 1812-5654, 10(4)(2010), 291-297.

[9] K. Florios, G. Mavrotas, and D. Diakoulaki, *Solving multiobjective, multiconstraint knapsack problems using mathematical programming and evolutionary algorithms*, European Journal of Operational Research, (2010) 14-21.

[10] L. Friedman, *A Competitive Bidding Strategy*, Operations Research, 4 (1956), 104-112.

[11] F. Glover, *Tabu Search Part II*, ORSA Journal on Computing 2(1)(1990), 4-32.

[12] Y. Guo, A. Lim, B. Rodrigues, and Y. Zhu, *Heuristics for a bidding problem*, Journal of Computers and Operations Research, Vol 33 No 8, (2006), 2179-2188.

[13] A. Holland, and B. O'sullivan, *Towards Fast Vickrey Pricing using Constraint Programming*, Artificial Intelligence Review, Vol 21 No 3-4 June (2004), 335-352.

[14] R.-C. Holte, *Combinatorial Auctions, Knapsack Problems and Hill-climbing Search.* In: Stroulia S. Matwin (Eds.), Advances in Artificial Intelligence, vol. 2056 of Lecture Notes in Computer Science, (2001), 57-66.

[15] H. Hoos, and C. Boutilier, *Solving combinatorial auctions using stochastic local search*, Proceedings of the 17th national conference on artificial intelligence, (2000), 22-29.

[16] H. Kellerer, U. Pferschy, and D. Pisinger, *Knapsack Problems*, Springer, Berlin Heidelberg New York, (2004).ISBN 3-540-40286-1.

[17] T. Kelly, *Generalized Knapsack Solvers for Multi-Unit Combinatorial Auctions: Analysis and Application to Computational Resource Allocation*, International Workshop on Agent-Mediated Electronic Commerce AMEC: Agent-Mediated Electronic Commerce VI. Theories for and Engineering of Distributed Mechanisms and Systems (2004), 73-86.

[18] M. Laumanns, L. Thiele, and E. Zitzler, *An adaptive scheme to generate the Pareto front based on the epsilon-constraint method*, In: Branke, J., Deb, K., Miettinen, K., Steuer, R. (Eds.), Practical Approaches to Multi-Objective Optimization, Dagstuhl Seminar Proceedings, vol. 04461. (2005), URN: urn:nbn:de:0030-drops-2465.

[19] M. Laumanns, L. Thiele, and E. Zitzler, *An efficient, adaptive parameter variation scheme for Metaheuristics based on the epsilon-constraint method*, European Journal of Operational Research 169, (2006), 932-942.
[20] S. Martello, and P. Toth, *Knapsack Problems*. Wiley, New York (1990).
[21] G. Mavrotas, and D. Diakoulaki, *A branch and bound algorithm for mixed zero one multiple objective linear programming*, European Journal of Operational Research 107, (1998), 530-541.
[22] G. Mavrotas, and D. Diakoulaki, *Multi-criteria branch and bound: A vector maximization algorithm for mixed zero one multiple objective linear programming*, Applied Mathematics and Computation 171, (2005), 53-71.
[23] T. Michalak, T. Rahwan, E. Elkind, M. Wooldrige, and N.-R. Jennings, *A hybrid exact algorithm for complete set partitionning*, Artificial Intelligence, vol. 230, (2016) 14-50.
[24] Y. Narahari, and P. Dayama, *Combinatorial auctions for electronic business*, Sadhana 30 (Pt. 2 & 3), (2005), 179-211.
[25] G.-L. Nemhauser, and L.-A. Wolsey, *Integer and Combinatorial Optimization*, Wiley (1999).
[26] N. Nisan, *Bidding and allocation in combinatorial auctions*, Proceedings of the 2nd ACM conference on Electronic commerce (EC-00), October (2000), 1-12, ACM New York, NY, USA.
[27] J. Pfeiffer, *The Winner Determination Problem in Multi-Unit Combinatorial and the Multi-Dimentional Knapsack Problem*, An Analysis of Optimization Methods. PhD thesis, University of Mannhein (2006).
[28] S.-J. Rassenti, L.-S. Vernon, and R.-L. Bulfin, *A Combinatorial Auction Mechanism for Airport Time Slot Allocation*, Bell Journal of Economics, 13, (1982), 402-417.
[29] M.-H. Rothkopf, *A Model of Rational Competitive Bidding*, Management Science, 15, (1969), 362-372.
[30] M.-H. Rothkopf, A. Pekee, and M. Ronald, *Computationally manageable combinatorial auctions*, Management Science, Vol 44 No 8 (1998), 1131-1147.
[31] T. Sandholm, and S. Suri, *Improved Optimal Algorithm for Combinatorial Auctions and Generalizations*, In: Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence, (2000), 90-97.
[32] T. Sandholm, S. Suri, A. Gilpin, and D. Levine, *CABOB: A Fast Optimal Algorithm for Combinatorial Auctions*, In: Proceedings of the International joint conferences on artificial intelligence, (2001), 1102-1108.
[33] T. Sandholm, *Optimal Winner Determination Algorithms*, P. Cramton et al. (ed.), Combinatorial Auctions, MIT Press (2006).
[34] S.-K. Shil, and S. Sadaoui, *Winner Determination in Multi-Objective Combinatorial Reverse Auctions*, 28th International Conference on Tools with Artificial Intelligence, (2016) IEEE.
[35] R.-E. Steuer, *Multiple Criteria Optimization Information Sciences: Theory, Computation and Application* , (ISBN 0-471-88846-X), Wiley Series on Probability and Mathematical Statistics-Applied, Wiley, New York, (1986), 546 pp.
[36] E.-L. Ulungu, *Optimisation combinatoire multicritère : détermination de l'ensemble des solutions efficaces et méthodes interactives*, PhD thesis, Faculté des Sciences, Université de Mons-Hainault, (1993).
[37] W. Vickrey, *Counterspeculation, Auctions, and Competitive Sealed Tenders*, Journal of Finance, 16 (1961), 8-37.
[38] Q. Wu, and J.-K. Hao, *A clique-based exact method for optimal winner determination in combinatorial auctions*, Information Sciences, vol 334 (2016), 103-121.

LaROMaD, Fac. Maths, USTHB, Pb 32, 16111 Bab Ezzouar, Algeria
E-mail address: adichechahra@yahoo.fr

LaROMaD, Fac. Maths, USTHB, Pb 32, 16111 Bab Ezzouar, Algeria
E-mail address: m-aider@usthb.dz