A *Min-Max* Tchebycheff based Local Search Approach for MOMKP

Imen Ben Mansour, Ines Alaya and Moncef Tagina
National School of Computer Sciences, COSMOS Laboratory, University of Manouba, Manouba, Tunis, 2010, Tunisia

Keywords: Multi-objective Multidimensional Knapsack Problem, Iterated Local Search, Scalarization Functions, Tchebycheff Functions.

Abstract: The multi-objective multidimensional knapsack problem (MOMKP) which is one of the hardest multi-objective combinatorial optimization problems, presents a formal model for many real world problems. Its main goal consists in selecting a subset of items in order to maximize m objective functions with respect to q resource constraints. For that purpose, we present in this paper a resolution approach based on a *Min-Max* **T**chebycheff iterated **L**ocal **S**earch algorithm called *Min-Max* TLS. In this approach, we propose designing a neighborhood structure employing a permutation process to exploit the most promising regions of the search space while considering the diversity of the population. Therefore, *Min-Max* TLS uses *Min-Max* $\mathcal{N}(s)$ as a neighborhood structure, combining a *Min-Extraction-Item* algorithm and a *Max-Insertion-Item* algorithm. Moreover, in *Min-Max* TLS two Tchebycheff functions, used as a selection process, are studied: the weighted Tchebycheff (WT) and the augmented weighted Tchebycheff (AugWT). Experimental results are carried out with nine well-known benchmark instances of MOMKP. Results have shown the efficiency of the proposed approach in comparison to other approaches.

1 INTRODUCTION

Many real-life optimization problems can hardly be formulated as a mono-objective problem, which explains the permanent growing interest in the field of multi-objective optimization. For instance, in transportation problem there is more than one objective to optimize. The cost of the transport, the duration and the capacity of the transport, all of this could be objectives to be optimized. In such problems, usually optimizing one objective, leads to degrading other objectives. Thus, finding a good trade-off between several conflicting objectives is one of the main goals of the multi-objective optimization problems (MOPs). So, it consists in optimizing simultaneously several conflicting objectives in order to find a set of solutions called Pareto front or set or non-dominated set.

The MOMKP modelizes many real world problems such as resource allocation (Shih, 2005), portfolio optimization (Penn et al., 1994) and budget allocation (Smeraldi and Malacaria, 2014). Moreover, MOMKP can be modelized as a subproblem such as the flight crew choice (Ehrgott and Ryan, 2002) and many other general integer programs. To solve the MOMKP, several approaches based on metaheuristics were proposed in the literature: In (Zitzler et al., 2002), Zitzler et al. introduced the well-known SPEA2 which is an elitist algorithm, based on a ranking dominance procedure. Deb et al. proposed the NSGAII (Deb et al., 2002), another well-known multi-objective evolutionary algorithm that is also an elitist approach and uses a different ranking dominance procedure. In (Lust and Teghem, 2008), a memetic algorithm integrates a tabu search method, called MEMOTS, was devised. A generic ACO algorithm (m-ACO) was presented and instantiated with four variants In (Alaya et al., 2007). m-ACO is parameterized by the number of ant colonies and the number of the pheromone structures. Recently, an indicator based ant colony approach which is abbreviated as IBACO is proposed in (BenMansour and Alaya, 2015). The algorithm uses the indicator optimization principle to guide ants during their search in the most promising areas by reinforcing the best solutions by rewarding pheromone.

Related Works. Local search approaches have been widely employed on the NP-hard multi-objective optimization problems, including problems formulated as a multi-objective multidimensional knapsack problem (Alsheddy and Tsang, 2009), (Vianna and Dianin, 2013) and(Liefooghe et al., 2013). Indeed, its simple and practical appearance has enabled it to be efficient in many real-world problem such as railway transportation, airline operations, portfolio optimiza-

Mansour, I., Alaya, I. and Tagina, M.
A *Min-Max* Tchebycheff based Local Search Approach for MOMKP.
DOI: 10.5220/0006433801400150
In *Proceedings of the 12th International Conference on Software Technologies (ICSOFT 2017)*, pages 140-150
ISBN: 978-989-758-262-2

tion and computer networks (Ehrgott and Gandibleux, 2004). Motivated by the success of these methods, several papers propose scalarization-based local search methods for MOMKP. These methods have received a great interest from the scientific community because of their success for solving multi-objective problems. They represent a simple way to transform a multi-objective problem into one single or a family of single objective optimization problems. In (Lust and Teghem, 2012), Lust and Teghem present the two-phase Pareto local search (2PPLS). The algorithm uses a Pareto-based local search combined with a very-large scale neighborhood method based on vectors weight. The 2PPLS combines Pareto local search and aggregation to solve MOMKP, it consists of two main phases. Phase 1 generates an approximation of all the supported solutions by solving a number of linear aggregation problems. Phase 2 applies a Pareto local search to every solution generated in Phase 1 to find non-supported Pareto optimal solutions. In (Alves and Almeida, 2007), a genetic algorithm based on Tchebycheff scalarization function called MOTGA has been proposed. The algorithm was applied to the MOMKP problem. The Pareto front is divided into a few small parts. Each part is approximated by a stage and each stage is guided by a different Tchebycheff aggregation function. More recently, a memetic algorithm based on decomposition (MOMAD) was proposed in (Ke et al., 2014). MOMAD combines the ideas of the two approaches: the 2PPLS and the evolutionary algorithm MOEA/D introduced in (Zhang and Li, 2007). In MOMAD, a Pareto local search method is first applied for the neighborhood search then a single objective local search is applied to each perturbed solution.

Motivations and Contributions. From the previously mentioned state-of-the-art algorithms, one can say that scalarization and weighted based methods present promising approaches to solve MOMKP. In the literature, various efforts have been directed toward the use of the metaheuristics to solve the MOMKP. Generally, these approaches present a good way to solve NP-hard problems. However, if a heuristic searching method is used these approaches may be very time consuming, especially for the large size instances. So, the challenge is optimizing both the quality results and the time-consuming. In this work, we propose an iterated multi-objective local search based on Tchebycheff function: *Min-Max* TLS for MOMKP. In order to investigate the impact of the choice of the Tchebycheff function on the selection process, the two well-known Tchebycheff functions are studied in this work: the weighted Tchebycheff: WT (Bowman, 1976) and the augmented weighted Tchebycheff: AugWT(Steuer and Choo, 1983). Moreover, one of the key aspects that has to be considered is the neighborhood structure since it plays a central role in a local search algorithm. *Min-Max* TLS integrates an efficient weighted neighborhood structure called *Min-Max* $\mathcal{N}(s)$ to explore the neighborhood. *Min-Max* $\mathcal{N}(s)$ allows to establish an order between items according to their profitability. It has the goal of finding the item that minimizes an extraction ratio to permute it with items that maximize an insertion ratio in order to improve the quality of the obtained solutions in a small computational time. Furthermore, in weighted based methods, the generation of the weight vectors presents a significant point and influences the quality of the generated solutions. Thus, *Min-Max* TLS defines a weight vector generation method called Gw: **G**radual **w**eight vector generation method. Gw creates a set of weight vectors, corresponding to search directions, guiding the search gradually on almost all regions of the Pareto front.

The paper is organized as follows. In the next section, we define the multi-objective optimization problems. In section 3, we present the multi-objective multidimensional knapsack problem. In section 4, the Tchebycheff functions used in this paper are described. Section 5 presents our main contributions, the proposed algorithm and its different main functions. Then, in section 6, we discuss the experimental results obtained from *Min-Max* TLS. Finally, in section 7, we end the paper with the conclusion and our perspectives.

2 MULTI-OBJECTIVE OPTIMIZATION PROBLEMS

Before introducing the MOMKP, let us introduce some useful notations and definitions related to a general multi-objective optimization problem.

Let X denote the decision space of a general optimization problem, Z the corresponding objective space and m objective functions f_1,...,f_m that assign to each decision vector $x \in X$ one objective vector $z=(f_1(x),...,f_m(x)) \in Z$. In the following, we assume that all objective functions are to be maximized:

- **Definition 2.1.** A solution z dominates a solution z', noted $z \succ z'$, iff $\forall\, i \in \{1,...,m\}$, $f_i(z) \geq f_i(z')$ and $\exists\, i \in \{1,...,m\}$,$f_i(z) > f_i(z')$;
- **Definition 2.2.** A solution z is called weakly non-dominated, if there exists no $z' \in Z$ such that for all $i \in \{1,...,m\}$,$f_i(z') > f_i(z)$;
- **Definition 2.3.** A solution z is Pareto-optimal or non-dominated, if a solution z' dominates z does

not exist.

The goal of a MOP is to find the set of all non-dominated solutions called Pareto set. When using a metheuristic approach, the goal is to find a Pareto set approximation.

3 MOMKP FORMULATION

The multi-objective multidimensional knapsack problem could be formulated as follows:

$$Maximize \sum_{j=1}^{n} p_j^k x_j \qquad k = 1,...,m \qquad (1)$$

$$Subject\,to \sum_{j=1}^{n} w_j^i x_j \leq b_i \qquad i = 1,...,q \qquad (2)$$

$$x_j \in \{0,...,1\} \qquad j = 1,...,n$$

where n is the number of items, for each item I_j is assigned a decision variable x_j equal to 1 if the item is selected, 0 otherwise. Each item I_j has a profit p_j^k relatively to the objective k and a weight w_j^i relatively to the resource i. The aim of the problem is to select a subset of items in order to maximize m objective functions while not exceeding q resource constraints. b_i is the total quantity available for the resource i.

In this paper, the considered instances assume that the number of objective functions is equal to the number of resources (m=q).

4 TCHEBYCHEFF FUNCTIONS

For decomposing a multi-objective optimization problem into one single-objective many scalarization functions exist. In following, we introduce two among the most commonly-used methods based on the Tchebycheff metric:

4.1 The Weighted Tchebycheff Function

$$WT(x|\lambda) = max_{k=1...m}\lambda^k|f_k(x) - r_k^*| \qquad (3)$$

where x is the solution to evaluate, λ=(λ^1...λ^m) is the weight vector, such that $\lambda^k \geq 0$ for all $k = 1...m$ and $\sum_{k=1}^{m}\lambda^k$=1. The ideal point $r^* = (r_1^*,...,r_m^*)$ is used as a reference point, calculated as follows:

$$r_k^* = max_{k=1...m} f_k(x) \qquad (4)$$

The weighted Tchebycheff in (3) is minimized to maximize each objective.

4.2 The Augmented Weighted Tchebycheff Function

One of the main advantage of the weighted Tchebycheff method is that by varying appropriately the weight vectors and/or the reference point, every non-dominated solution of the Pareto front of the studied MOP, explicitly includes non-convex and discrete problems, can be found (Steuer, 1986). On the other hand, the weighted Tchebycheff method generates also the weakly non-dominated solutions which is often undesirable in MOP. In (Steuer and Choo, 1983), Steuer and Choo suggested to add an l_1-term, parameterized by ε, to $WT(x|\lambda)$ that helps to avoid the weakly non-dominated solutions. The resulting function is defined as follows:

$$AugWT(x|\lambda) = max_{k=1...m}\lambda^k|f_k(x) - r_k^*| + \varepsilon\sum_{k=1}^{m}\lambda^k|f_k(x) - r_k^*| \qquad (5)$$

where $\varepsilon \geq 0$ is usually chosen as a small positive number and r^* is calculated as defined in Eq. (4).

5 OUR PROPOSED APPROACH *MIN-MAX* TLS: *MIN-MAX* TCHEBYCHEFF BASED LOCAL SEARCH

Let $\mathcal{P}$ be the current population of *Min-Max* TLS of size N. Initially, a weight vector is selected from the set $\mathcal{V}$ of H weight vectors, generated according to the proposed Gw method. Then, for each solution s of $\mathcal{P}$ the neighborhood is explored following the proposed weighted neighborhood structure *Min-Max* $\mathcal{N}(s)$, until a good solution is found i.e. one which is better than at least the worst solution w in $\mathcal{P}$ in terms of the used Tchebycheff function. This process is iterated until all solutions in $\mathcal{P}$ are explored. All the non-dominated solutions found during the local search step are stored in the archive $\mathcal{A}$. The algorithm stops when a maximum number of iterations $Tmax$ is met and return the archive $\mathcal{A}$. The algorithm 1 shows a detailed description of the *Min-Max* TLS algorithm.

5.1 Weight Vectors Generation

In order to generate the set $\mathcal{V}$, we propose the Gw method: Let λ=(λ^1...λ^m) be the weight vector, such that $\lambda^k \geq 0$ for all $k = 1...m$ and $\sum_{k=1}^{m}\lambda^k$=1. For the bi-objective case, the weight vectors are calculated as

Algorithm 1: *Min-Max* TLS algorithm.

Begin
  $\mathcal{V} \leftarrow$ *Generate Weight Vectors* (H)
repeat
    $\mathcal{P} \leftarrow$ *Perturbation*$(\mathcal{P}, N)$
    $\mathcal{A} \leftarrow$ *Non-dominated solutions*$(\mathcal{P})$
    $\lambda \leftarrow$ *SelectWeightVector* $(\mathcal{V}, H)$
    for all $s \in \mathcal{P}$ **do**
      repeat
        UpdateReferencePoint
        $s^* \leftarrow$ *Neighborhood*(s)
        if *Acceptance*$(s^*, w, \mathcal{P})$ **then**
          $\mathcal{P}' \leftarrow$ *Replace*$(s^*, w, \mathcal{P})$
        end if
      until $s^* \neq w$
    end for
    $\mathcal{A} \leftarrow$ *Non-dominated solutions*$(\mathcal{A} \cup \mathcal{P}')$
until $Tmax$ **is reached**
End

follow:

$$\lambda^1(t) = \ln\left[\left(\frac{4*t*e}{FQ}\right) + \cos\left(\frac{2*\pi*t}{FQ}\right)\right] \quad (6)$$

$$\lambda^2(t) = 1.0 - \lambda^1(t) \quad (7)$$

where t is the iteration index and e is exp (1). The weight vectors are controlled by FQ, which is the weight change frequency. If the FQ is well set, the population can move smoothly from one point to another thus the whole regions can be explored and almost all points covering the Pareto front can be found through this appropriate variation of research orientations. Fig. 1 shows an example of how the weight vector values λ^1 and λ^2 change within 200 iterations during the local search process.

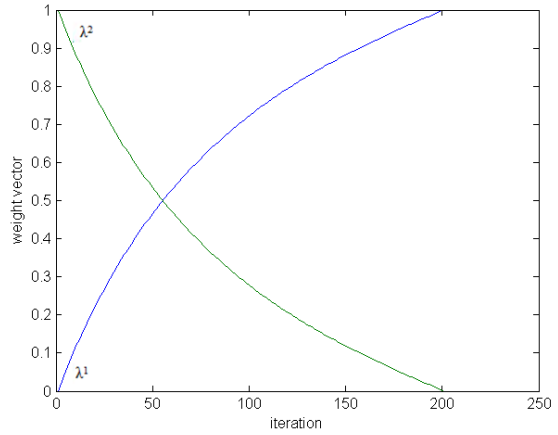

Figure 1: The weight vector value change within 200 iterations.

The extension of the Gw method with more than two objectives is theoretically straightforward. The algorithm 2 shows the Gw, in case of m=4 objectives.

Algorithm 2: Gw algorithm.

Begin
for i *from* 0 *to* $FQ/4$ **do**
    λ^1=$\ln\left[\left(\frac{4*i*e}{FQ}\right) + \cos\left(\frac{2*\pi*i}{FQ}\right)\right]$
    for j *from* 0 *to* $FQ/4$ **do**
      λ^2=$(1.0-\lambda^1)*\ln\left[\left(\frac{4*j*e}{FQ}\right) + \cos\left(\frac{2*\pi*j}{FQ}\right)\right]$
      for k *from* 0 *to* $FQ/4$ **do**
        λ^3=$(1.0-\lambda^1-\lambda^2)*$
        $\ln\left[\left(\frac{4*k*e}{FQ}\right) + \cos\left(\frac{2*\pi*k}{FQ}\right)\right]$
        λ^4=$1.0-\lambda^1-\lambda^2-\lambda^3$
      end for
    end for
end for
End

The algorithm 2 is executed only once before the beginning of the main loop of *Min-Max* TLS. Once the initial population is generated, a weight vector is selected from the set $\mathcal{V}$, according to its order of generation by the Gw method and assigned to the solution to be evaluated in such way that all the member of the population have the same weight vector during the same iteration.

5.2 Initial Population Initialization

Like in most evolutionary approaches, the first initial population in *Min-Max* TLS is randomly created. In *Min-Max* TLS, a solution s is composed of two subsets I_l^+=$\{I_1^+, I_2^+, ..., I_T^+\}$ and $I_{\bar{l}}^-$=$\{I_1^-, I_2^-, ..., I_{NT}^-\}$ where I_l^+ is the subset of the items selected in the solution s, T is the number of taken items, $I_{\bar{l}}^-$ is the subset of the remaining items (unselected items) and NT is the number of untaken items. To create a first random population, *Min-Max* TLS creates N random solutions. Every solution s is created by selecting a random item j among the n items and added it to the solution s. If any resource capacity is violated by an item j, the item is placed in the subset $I_{\bar{l}}^-$ otherwise the item is added to the subset I_l^+. This process is iterated until all items are placed.

5.3 Perturbation

After the update of the archive $\mathcal{A}$, a new population is generated. In order to reach different and new region of the search space, where the exploration starts in the

next local search step, we propose to use a perturbation function to generate the new population: At each iteration, N new solutions are randomly selected from the archive $\mathcal{A}$ if the size of $\mathcal{A}$ exceeds N, else if the archive size is less than the size of the population, the missing solutions are created randomly as done in the first initial population. Let μ be the rate of noise, for every solution s we remove randomly $(\mu * T)$ items from the subset I_l^+ and we add them to the subset $I_{\bar{l}}^-$ of the solution s. Then, as long as we do not violate any resource constraints. Items are randomly added from $I_{\bar{l}}^-$ to I_l^+. In fact, this random move allows us to reach new local optima. They can be weak or strong noise which means moving to a near local optima or jump totally to a new region where it is considerably difficult to find a new local optima.

5.4 Update Reference Point

When a new solution is introduced in the population, the reference point r^* is updated. After the initialization process of the population and after each neighborhood exploration, the new maximal value of all objective functions is calculated according to Eq. (4).

5.5 Neighborhood Structure

The generation of the neighborhood is one of the most important part of a local search algorithm. In this paper, we use a proposed weighted neighborhood structure *Min-Max* $\mathcal{N}(s)$. By means of the two proposed algorithms: *Min-Extraction-Item* algorithm and *Max-Insertion-Item* algorithm, *Min-Max* $\mathcal{N}(s)$ tries to gives a better neighbor of a solution s.

5.5.1 *Min-Extraction-Item*

For each taken item I_l^+ the ratio $U(l^+)$ is calculated as follows:

$$U(l^+) = \frac{\sum_{k=1}^{m} \lambda^k(t) p_{l^+}^k}{\sum_{i=1}^{q} w_{l^+}^i} \tag{8}$$

where $\lambda^k(t)$ is the weight vector selected from the set $\mathcal{V}$, $\sum_{k=1}^{m} \lambda^k(t) p_{l^+}^k$ is the weighted profit of the item I_l^+ and $\sum_{i=1}^{q} w_{l^+}^i$ its overall weight. The ratio $U(l^+)$ measures the utility value of each item, the lower this ratio is, the worst the item is.

Once the ratio $U(l^+)$ is calculated for all items I_l^+=$\{I_1^+, I_2^+, ..., I_T^+\}$, a list L_U containing all the items I_l^+ is created and sorted in ascending order according to the ratio $U(l^+)$ of each item. So that the worst items that minimize the ratio $U(l^+)$ are placed on the top of the list. The algorithm 3 outlined the *Min-Extraction-Item* algorithm. This algorithm is executed only when a new solution s is selected for neighborhood exploration.

Algorithm 3: *Min-Extraction-Item* algorithm.

```
1: Begin
2: for l+ from 1 to T do
3:     Compute U(l+)
4:     Add Item I_l+ to L_U
5: end for
6: Sort L_U in ascending order
7: End
```

5.5.2 *Max-Insertion-Item*

For each untaken item $I_{\bar{l}}^-$ the ratio $\overline{U}(\bar{l})$, inspired from (Alaya et al., 2004), is calculated. We define $R_i(s)$=b_i-$\sum_{l=1}^{T} w_l^i$ as the remaining quantity of the resource i after the construction of the solution s. The ratio $\overline{U}(\bar{l})$ of an item $I_{\bar{l}}^-$ can be calculated as follows:

$$\overline{U}(\bar{l}) = \frac{\sum_{k=1}^{m} \lambda^k(t) p_{\bar{l}}^k}{\sum_{i=1}^{q} \left(\frac{w_{\bar{l}}^i}{R_i(s)} \right)} \tag{9}$$

where $p_{\bar{l}}^k$ and $w_{\bar{l}}^i$ are respectively the profit and the weight of the candidate item. The ratio $\overline{U}(\bar{l})$ measures the quality of the candidate item according to the solution s where the higher this ratio is, the better the item is.

As done with the taken items in the *Min-Extraction-Item* algorithm, a list $L_{\overline{U}}$ containing all the candidate items $I_{\bar{l}}^-$ is created and sorted in decreasing order according to the ratio $\overline{U}(\bar{l})$ of each item. So that the most profitable items that maximize the ratio $\overline{U}(\bar{l})$ are placed on the top of the list. The algorithm 4 outlines the *Max-Insertion-Item* algorithm.

Algorithm 4: *Max-Insertion-Item* algorithm.

```
1: Begin
2: for l̄ from 1 to NT do
3:     Compute Ū(l̄)
4:     Add Item I_l̄^- to L_Ū
5: end for
6: Sort L_Ū in decreasing order
7: End
```

5.5.3 *Min-Max* $\mathcal{N}(s)$

A neighbor s^* is created by removing an item from the list L_U and adding items from the list $L_{\overline{U}}$ as long as no resource constraint is violated. The algorithm 5 describes the generation of a neighbor of a solution s.

Algorithm 5: *Min-Max* $\mathcal{N}(s)$.

Begin
Min-Extraction-Item(T)
while s^* **is the worst do**
    Max-Insertion-Item(NT)
    $s^* \leftarrow$ *Remove* $L_U(l^+)$ *from* s
    while no constraint is violated do
        $s^* \leftarrow$ *Add* $L_{\overline{U}}(\bar{l})$ *to solution* s
    end while
end while
End

5.6 Acceptance Criterion and Replacement Function

In the local search algorithms, the neighborhood exploration stops at the first better solution or when the entire neighborhood is explored. In this work, we choose to stop the exploration once the first better neighbor is found relatively to the used Tchebycheff function. This choice is guided by one main reason is that this mechanism allows to speed up the convergence of the population.

The neighbor s^* is accepted in the population if its fitness value, calculated according to the Weighted Tchebycheff function (Eq. 3) or the Augmented Weighted Tchebycheff function (Eq. 5), is better than the worst solution w in the current population. If so, s^* replaces the worst solution w. Otherwise the neighbor is rejected, the next item on the list L_U is selected, i.e. the item to be removed from the solution s, and the algorithm 5 is executed again. This process is iterated until a neighbor s^* better than at least the worst solution w is found.

6 EXPERIMENTAL RESULTS

The experiments carried out consider two versions of *Min-Max* TLS: *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT} where the solutions are evaluated using respectively the Weighted Tchebycheff function and the Augmented Weighted Tchebycheff function. In order to evaluate the quality of the proposed algorithm and to have a fair comparison, we choose to compare the two proposed versions of *Min-Max* TLS with scalarizition-local search-based methods : the MOMAD (Ke et al., 2014), the 2PPLS algorithm (Lust and Teghem, 2012) and the MOTGA (Alves and Almeida, 2007). For both of the two versions as well as the compared algorithms, we considered the benchmark instances defined in (Zitzler and Thiele, 1999), that are widely used in testing several multi-objective heuristics, containing nine instances with 250, 500 and 750 items, in combination with 2, 3 and 4 objectives.

6.1 Experimental Setup

The used parameter settings for *Min-Max* TLS are adjusted experimentally and presented as follow: The population size N is set to 10, we choose a small rate noise μ=0.05 to apply on selected solutions in order to focus the search around the most useful solutions. The parameter ε used in the AugWT function is set to 10^{-3} as suggested in (Steuer, 1986), to ensure that weakly non-dominated solutions are avoided and that all non-dominated solutions of the studied (potentially non-convex) problem can be found. The change frequency FQ is set according to the Table 1. In order to let the weight vectors change smoothly from one iteration to another and so the whole Pareto front will be explored, the number of generated weight vectors H and the stopping criterion $Tmax$ is set to $FQ/4^{m-1}$. The MOMAD, 2PPLS and MOTGA were configured as recommended by their authors, for each algorithm and each instance 30 runs are considered.

Table 1: The change frequency FQ Setting.

Number of Objectives	FQ
2	800
3	40
4	20

6.2 Performance Metrics

In order to evaluate the results generated by the different algorithms, we used three measures:

- The hypervolume Difference: which computes the difference between a reference set, which corresponds to the set of the non-dominated solutions obtained from the union of all solutions of the different runs, and the set of solutions to be evaluated in terms of hypervolume as defined in (Zitzler and Thiele, 1999). The obtained values have to be as close as possible to zero to prove the efficiency of an algorithm against the other approaches;
- The Mann-Whitney Statistical Test: (Knowles et al., 2005) is applied to the results obtained by the hypervolume difference in order to prove that the difference between two algorithms are statistically significant. We reject the null-hypothesis "no significant difference between the two algorithms A and B" if the P-value is lower than the significance level 5%;

- The summary Attainment Surface: (Grunert Da Fonseca et al., 2001) is used to compare the approach behaviors in a graphical way. The attainment surface is defined by the objective vectors that have been attained by at least one approximation set of the tested algorithm. In the following we compare the median (50%) attainment surface of the fronts non-dominated solutions found for 30 runs of all the compared algorithms.

All the computational results were obtained by the performance assessment software package PISA provided in http://www.tik.ee.ethz.ch/sop/pisa/.

6.3 Comparison Results

We performed 30 independent runs of each compared algorithm on all the tested instances. In order to have a good idea of the overall performance of each approach, the average hypervolume is computed for each algorithm and each instance. The hypervolume difference results of the proposed algorithms *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT}, and the compared algorithms MOMAD, 2PPLS and MOTGA appear in Table 2. Moreover, the Table 2 represents the results obtained by the Mann-Whitney statistical test. The M-W.T column contains the number of the algorithm which is statistically outperformed by the corresponding algorithm. For example, on the 2x250 instance, *Min-Max* TLS_{WT} statistically outperformed the algorithm number 3 and the algorithm number 5 which correspond to the MOMAD algorithm and the MOTGA algorithm respectively. "_" means that the corresponding algorithm does not statistically outperform any other algorithm, as is the case with 2PPLS on the 2x750 instance. *m*x*n* denotes the tested instance, *m* is the number of objectives, *n* is the number of items and (N/A) denotes that the results are not available.

By analyzing the Table 2, it is clear that our proposed approach performs better than the other compared algorithms. In fact, *Min-Min* TLS gives better hypervolume difference results than MOMAD on all the tested instances and better than MOTGA in eight out of nine instances. Moreover, it seems that the average results of MOMAD and MOTGA decrease according to the size of the problem. For the largest and hardest instances, the quality of the obtained results of MOMAD and MOTGA decease clearly which is not the case of *Min-Max* TLS. Thus, the difference between the compared approaches became much more important with the largest instances. By comparing *Min-Max* TLS with 2PPLS on the three bi-objective instances, the two approaches obtained about the same hypervolume values except on the 2x500 instance where 2PPLS preforms better than *Min-Max* TLS. By analyzing the results of the Mann-Whitney statistical test shown in the Table 2, the first important information that we can extract from the table is that *Min-Max* TLS outperforms significantly MOMAD and MOTGA on all the largest and hardest instances with 3 and 4 objectives. However, MOMAD and MOTGA never significantly outperform the results returned by the proposed approach on these instances. From the three bi-objective instances, we can observe that *Min-Max* TLS outperforms significantly MOMAD and MOTGA on 2x750 and 2x250 instances. On the 2x500 instance, *Min-Max* TLS outperforms only MOMAD.

Fig. 2 compares the average hypervolume difference of *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT} according to the number of objectives. This figure shows clearly the behaviors of the two versions. As it was previously remarked, the hypervolume values of *Min-Max* TLS_{WT} are slightly better than *Min-Max* TLS_{AugWT} on the smallest instances (2 and 3 objectives with 250 and 2 objectives with 500 items) while with the largest instances with 4 objectives and with 500 and 750 items, *Min-Max* TLS_{AugWT} find slightly better values than *Min-Max* TLS_{WT}. As a conclusion, one can say that, according to the Table 2 and figure 2, *Min-Max* TLS_{AugWT} slightly outperforms *Min-Max* TLS_{WT} for 5 out of the 9 instances in terms of the average hypervolume difference but statistically there is no significant difference between the two versions: *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT}.

Table 3 gives the average computational time consumed by each algorithm in seconds. From the table, it is clear that our proposed approach *Min-Max* TLS consumes significant shorter CPU time than the other approaches. In fact, the time consumed by *Min-Max* TLS is shorter than the other approaches for some instances about 8 times. When comparing the CPU time of *Min-Max* TLS_{WT} against *Min-Max* TLS_{AugWT}, one can observe that the two algorithms consume almost the same running time. Nevertheless, *Min-Max* TLS_{AugWT} consumes slightly more running time than *Min-Max* TLS_{WT} on the same instances.

Fig. 3 shows the median attainment surfaces of the approximation sets returned by the two versions *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT}, MOMAD, 2PPLS and MOTGA for the bi-objectives instances 2x250, 2x500 and 2x750. The first important remarque that can be observed from this figure is that all the tested approaches provide a well-distributed Pareto front where the obtained points cover almost all the Pareto front. For the instance containing 250 items, the surfaces are almost confused, it is difficult to distinguish the algorithms. For the

Table 2: Hypervolume Difference Average Values and Mann-Whitney Statistical Test Results.

Instances	*Min-Max* TLS$_{WT}$		*Min-Max* TLS$_{AugWT}$		MOMAD		2PPLS		MOTGA	
Algorithm Number	1		2		3		4		5	
	Avg	M-W.T	Avg	M-W.T	Avg	M-W.T	Avg	M-W.T	Avg	M-W.T
2x250	2.35E-01	3,5	2.52E-01	3	3.16E-01	_	2.32E-01	_	2.72E-01	_
2x500	1.93E-01	3	2.16E-01	3	3.56E-01	_	1.20E-01	1,2	1.41E-01	1,2
2x750	2.14E-01	3,5	2.13E-01	3,5	3.00E-01	_	2.10E-01	_	2.90E-01	_
3x250	2.12E-01	3,5	2.21E-01	3,5	3.93E-01	_	N/A		3.29E-01	_
3x500	2.27E-01	3,5	2.16E-01	3,5	4.56E-01	_	N/A		3.96E-01	_
3x750	1.91E-01	3,5	1.96E-01	3,5	4.31E-01	_	N/A		2.90E-01	_
4x250	2.20E-01	3,5	2.18E-01	3,5	3.76E-01	_	N/A		3.79E01	_
4x500	1.93E-01	3,5	1.84E-01	3,5	3.97E-01	_	N/A		3.97E-01	_
4x750	1.88E-01	3,5	1.76E-01	3,5	4.59E-01	_	N/A		4.27E-01	_

Table 3: Average CPU Time of *Min-Max* TLS$_{WT}$, *Min-Max* TLS$_{AugWT}$, MOMAD, 2PPLS and MOTGA in seconds.

Instances	*Min-Max* TLS$_{WT}$	*Min-Max* TLS$_{AugWT}$	MOMAD	2PPLS	MOTGA
2x250	0.6	0.8	5.2	3.1	0.9
2x500	1.9	2.3	15.5	14.8	3.3
2x750	4.0	4.3	23.5	25.1	9.6
3x250	1.0	1.0	7.5	N/A	8.8
3x500	3.4	3.7	19.1	N/A	15.2
3x750	5.8	6.4	35.7	N/A	40.7
4x250	4.8	5.0	10.7	N/A	8.6
4x500	13.6	16.1	25.5	N/A	27.4
4x750	24.3	28.8	45.7	N/A	43.2

other instances, the surfaces of MOMAD, 2PPLS and MOTGA are slightly above the surfaces returned by *Min-Max* TLS$_{WT}$ and *Min-Max* TLS$_{AugWT}$ on a small middle region of the Pareto front. While throughout the extremity, both of the surfaces of *Min-Max* TLS$_{WT}$ and *Min-Max* TLS$_{AugWT}$ are close to those of MOMAD, 2PPLS and MOTGA. We note that there is no clear difference between the surfaces obtained by *Min-Max* TLS$_{WT}$ and *Min-Max* TLS$_{AugWT}$. Thus, this graphic confirms the numerical results obtained previously, where we have found that, generally, for the bi-objective instances the compared algorithms are not significantly different.

6.4 Discussion

As a conclusion of this experimentation section, the results have shown that the proposed approach performs better than the compared approaches. When evaluating the *Min-Max* TLS algorithm for solving the MOMKP, some useful conclusions can be extracted. First, the *Min-Max* TLS performs statistically better especiallly for the largest and hardest instances where the best values for the instances with 3 and 4 objectives are found by the two versions of *Min-Max* TLS. Second, the difference between the two versions *Min-Max* TLS$_{WT}$ and *Min-Max* TLS$_{AugWT}$ is very small. *Min-Max* TLS performs slightly better using *AugWT* function. Third, *Min-Max* TLS$_{AugWT}$ needs more computational time than *Min-Max* TLS$_{WT}$. In fact, the augmented weighted Tchebycheff version consumes slightly more CPU time than the weighted Tchebycheff version, especially with the largest instances but produces better results.

Finally, let us mention that the efficiency of the proposed approach is due to many factors: the proposed Gw method, the perturbation function, the neighborhood structure, the acceptance criterion and the replacement strategy. The Gw method has a significant role in *Min-Max* TLS. Like any weighted approach, the selection of the weights can lead to a better performance. In fact, the Gw method tries to appropriately varying the search directions, all the member of population attempt to target almost all parts on the Pareto front. Hence, different Pareto-optimal points can be obtained. Furthermore, the initialization of the local search population is a very important function. The used perturbation function allows to generate a new population using information about the good solutions obtained during the previous iterations. Also, the neighborhood structure is a crucial part of the local search algorithm. Here, the proposed *Min-Max* $\mathcal{N}(s)$ tries to remove the least valuable item and replace it with the most profitable items according to the current solution. Therefore, it provides an efficient way to speed up the search while leading the algorithm to converge. Lastly, the acceptance criterion and the replacement strategy. The used replacement strategy in this work is a convergence-promoting mechanism, it favors exploitation. While the acceptance criterion, which can be considered as a diversity-promoting mechanism, favors more the ex-

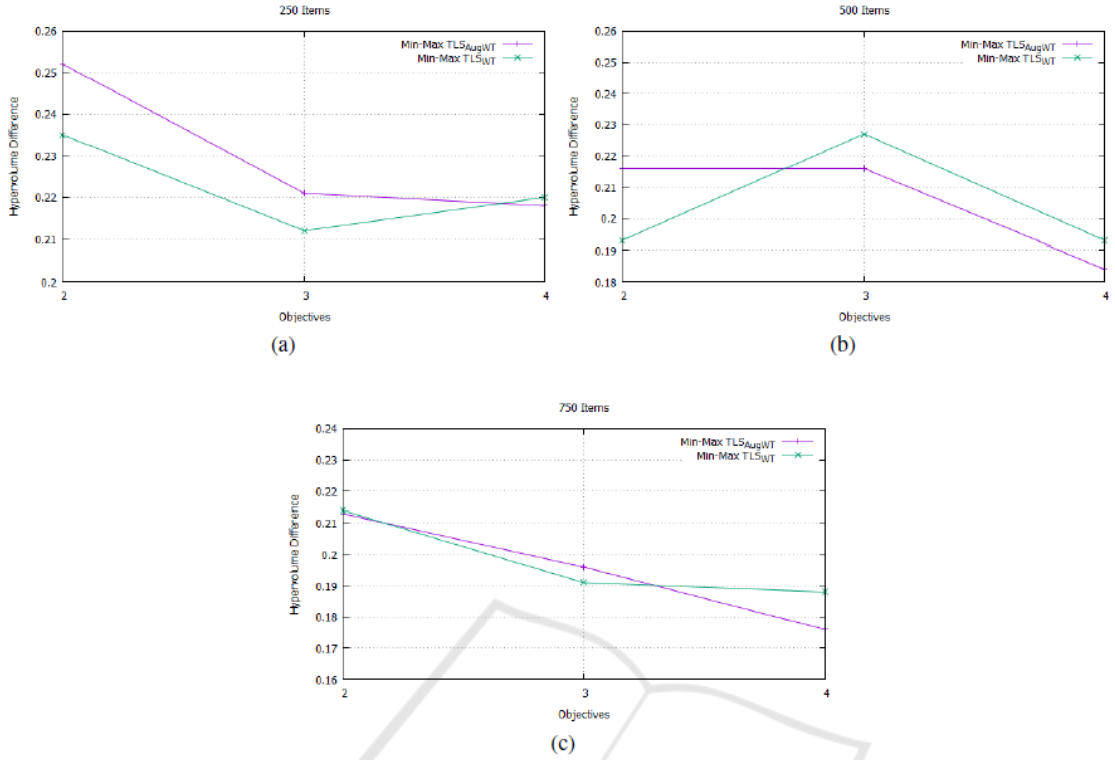

Figure 2: Average Hypervolume Difference of *Min-Max* TLS_{WT} and *Min-Max* TLS_{AugWT} with (a) 250, (b) 500 and (c) 750 Items.

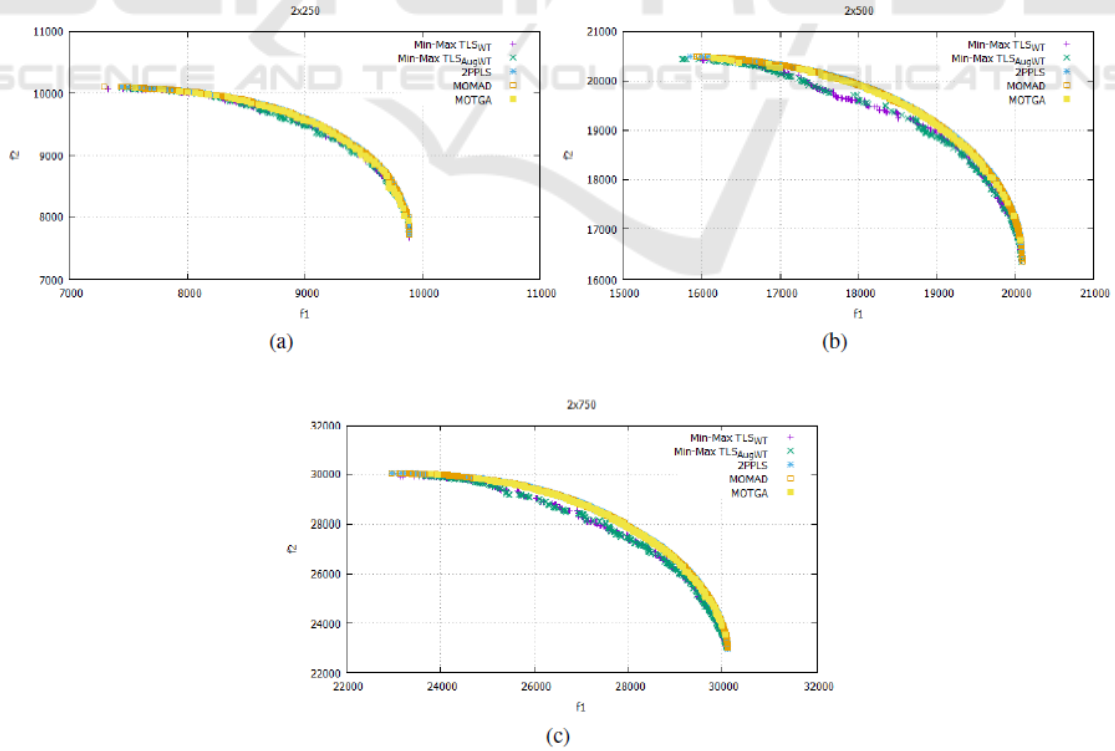

Figure 3: Illustration of the Median Attainement Surfaces Obtained by *Min-Max* TLS_{WT}, *Min-Max* TLS_{AugWT}, MOMAD, 2PPLS, MOTGA with (a) 2x250, (b) 2x500 and (c) 2x750 Instances.

ploration. Thus, both of them help to provide a good balance between exploration and exploitation during the search process.

7 CONCLUSION AND PERSPECTIVES

In this paper, we have presented a new multi-objective local search using the scalarization concept. The proposed approach *Min-Max* TLS has proved its efficiency for solving the multi-objective multidimensional knapsack problem in comparison with three of the well-known state-of-the-art algorithms. In fact, *Min-Max* TLS is significantly better than the compared algorithms on almost all the tested instances. In addition, experimental results have shown the performance of the two proposed versions of *Min-Max* TLS on a short processing time. As improvement of this work, one potential direction of future research would be to combine *Min-Max* TLS with another metaheuristic method as the ant colony approach. This metaheuristic could replace the initial population function, since the use of an ant colony approach combined with a local search method is known to perform well on many problems. Another perspective of this work is to apply this approach for solving other multi-objective optimization problems. In fact, it would be interesting to adapt *Min-Max* TLS to other optimization problem to investigate its scalability and to have an overview of its efficiency.

ACKNOWLEGMENT

We are grateful to the anonymous reviewers for their insightful comments to improve our paper.

REFERENCES

Alaya, I., Solnon, C., and Ghédira, K. (2004). Ant algorithm for the multi-dimensional knapsack problem. *Proceedings of International Conference on Bioinspired Optimization Methods and their Applications (BIOMA)*, 1:63–72.

Alaya, I., Solnon, C., and Ghédira, K. (2007). Ant colony optimization for multi-objective optimization problems. *19th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'07)*, 1:450–457.

Alsheddy, A. and Tsang, E. (2009). Guided pareto local search and its application to the 0/1 multi-objective knapsack problems. *Proceedings of the eighth metaheuristic international conference (MIC09)*.

Alves, M. J. and Almeida, M. (2007). MOTGA: A multiobjective tchebycheff based genetic algorithm for the multidimensional knapsack problem. *Computers & OR*, 34(11):3458–3470.

BenMansour, I. and Alaya, I. (2015). Indicator based ant colony optimization for multi-objective knapsack problem. *Knowledge-Based and Intelligent Information & Engineering Systems 19th Annual Conference*, 60:448–457.

Bowman, V. J. (1976). On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. *In H. Thieriez and S. Zionts, editors, Multiple Criteria Decision Making*, 1:76–85.

Deb, K., Pratap, A., S.Agarwal, and Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transaction on Evolutionary Computation*, 6(2):181–197.

Ehrgott, M. and Gandibleux, X. (2004). Approximative solution methods for multiobjective combinatorial optimization. *Top*, 12:1–63.

Ehrgott, M. and Ryan, D. M. (2002). Constructing robust crew schedules with bicriteria optimization. *Journal of Multi-Criteria Decision Analysis*, 11(3):139–150.

Grunert Da Fonseca, V., Fonseca, C. M., and Hall, A. O. (2001). Inferential performance assessment of stochastic optimisers and the attainment function. *1st International Conference on Evolutionary Multi-criterion Optimization (EMO 2001)Lecture Note in Computer Science, Springer*, pages 213–225.

Ke, L., Zhang, Q., and Battiti, R. (2014). A simple yet efficient multiobjective combinatorial optimization method using decompostion and pareto local search. *IEEE Trans on Cybernetics*.

Knowles, J. D., Thiele, L., and Zitzler, E. (2005). A tutorial on the performance assessment of stochastive multi-objective optimizers. *Technical report TIK-Report*.

Liefooghe, A., Paquete, L., and Figueira, J. (2013). On local search for bi-objective knapsack problems. *Evol. Comput.*, 21(1):179–196.

Lust, T. and Teghem, J. (2008). Memots: a memetic algorithm integrating tabu search for combinatorial multiobjective optimization. *RAIRO - Operations Research*, 42:3–33.

Lust, T. and Teghem, J. (2012). The multiobjective multidimensional knapsack problem: a survey and a new approach. *International Transactions in Operational Research*, 19:495–520.

Penn, M., Hasson, D., and Avriel, M. (1994). Solving the 0/1 proportional knapsack problem by sampling. *J. Optim Theory Appl.*, 80:261–272.

Shih, H. (2005). Fuzzy approach to multilevel knapsack problems. *Computers and Mathematics with Applications*, 49:1157–1176.

Smeraldi, F. and Malacaria, P. (2014). How to spend it: Optimal investment for cyber security. *Proceedings of the 1st International Workshop on Agents and Cyber-Security*.

Steuer, R. E. (1986). *Multiple Criteria Optimization: Theory, Computation and Application*. John Wiley, New York.

Steuer, R. E. and Choo, E. U. (1983). An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical Programming*, pages 326–344.

Vianna, D. S. and Dianin, M. F. (2013). Local search based heuristics for the multiobjective multidimensional knapsack problem. *Production journal*, 1:478–487.

Zhang, Q. and Li, H. (2007). MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions Evolutionary Computation*, 11:712–731.

Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: improving the strength pareto evolutionary algorithm for multiobjective optimization. In Giannakoglou, K. et al., editors, *Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (EUROGEN 2001)*, volume 1, pages 95–100. International Center for Numerical Methods in Engineering (CIMNE).

Zitzler, E. and Thiele, L. (1999). Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 1:257–271.