A Hybrid Evolutionary Metaheuristics (HEMH) applied on 0/1 Multiobjective Knapsack Problems

Ahmed Kafafy

ahmedkafafy80@gmail.com

Ahmed Bounekkar

bounekkar@univ-lyon1.fr

Stéphane Bonnevay

stephane.bonnevay@univ-lyon1.fr

Université Claude Bernard Lyon 1- Laboratoire ERIC
Ecole Polytechnique Universitaire, 15 Boulevard Latarjet, 69622 Villeurbanne cedex, France.

ABSTRACT

Handling Multiobjective Optimization Problems (MOOP) using Hybrid Metaheuristics represents a promising and interest area of research. In this paper, a Hybrid Evolutionary Metaheuristics (HEMH) is presented. It combines different metaheuristics integrated with each other to enhance the search capabilities. It improves both of intensification and diversification toward the preferred solutions and concentrates the search efforts to investigate the promising regions in the search space. In the proposed HEMH, the search process is divided into two phases. In the first one, the DM-GRASP is applied to obtain an initial set of high quality solutions dispersed along the Pareto front. Then, the search efforts are intensified on the promising regions around these solutions through the second phase. The greedy randomized path-relinking with local search or reproduction operators are applied to improve the quality and to guide the search to explore the non discovered regions in the search space. The two phases are combined with a suitable evolutionary framework supporting the integration and cooperation. Moreover, the efficient solutions explored over the search are collected in an external archive. The HEMH is verified and tested against some of the state of the art MOEAs using a set of MOKSP instances commonly used in the literature. The experimental results indicate that the HEMH is highly competitive and can be considered as a viable alternative.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods and Search – *heuristic methods*.
G.1.6 [**Numerical Analysis**]: optimization- *constrained optimization, global optimization, integer programming*.

General Terms

Algorithms, Experimentation, Performance, Verification.

Keywords:

Data Mining, Evolutionary Algorithm, GRASP, Hybridization, Metaheuristics, Multiobjective Optimization, Path-relinking.

1. INTRODUCTION

Many of the real-world problems can be modeled as Multiobjective Combinatorial Optimization Problems (MOCOP), which are often characterized by their large size and the presence

of multiple, conflicting objectives. In general, the basic task in multiobjective optimization is the identification of the set of Pareto optimal solutions or even a good approximation set to the Pareto Front (*PF*). Despite the progress in solving MOCOP exactly, the large size often means that Metaheuristics (MH) are required for their solution in reasonable time. Many of MHs have been introduced in the last thirty years [7] such as Evolutionary Algorithms (EA), Simulated Annealing (SA), Tabu Search (TS), Scatter Search (SS), Path-Relinking, Iterated Local Search (ILS), Guided Local Search (GLS), Particle Swarm Optimization (PSO) and Greedy Randomized Adaptive Search Procedure (GRASP)…etc. More details are found in [1].

Multiobjective Evolutionary Algorithms (MOEAs) are a very active research area. They have recently received increase interest because they offer practical advantages in facing difficult optimization problems. Solving MOOPs and their applications using evolutionary algorithms have been investigated by many authors [3] [5] [10] [22] [24]. NSGAII [3] and SEPA2 [24] are the most popular Pareto dominance based MOEAs that have been dominantly used. Based on many traditional mathematical programming methods for approximating the *PF* [14], the approximation of the *PF* can be decomposed into a number of single objective subproblems. Some of the MOEAs adopt this idea such as MOGLS [11], MOEA/D [21]. Many of the search algorithms attempt to obtain the best from a set of different MHs that perform together, complement each other and augment their exploration capabilities. They are commonly called *Hybrid* MH. Diversification and intensification [1] are the two major issues when designing a global search method. Diversification refers to the ability to visit many and different regions in the search space, while intensification refers to the ability to obtain high quality solutions within those regions. A search algorithm must balance between sometimes-conflicting two goals. The design of *Hybrid* MH can give the ability to control this balance [13].

This paper tends to study the hybridization of different MHs and analyze its effect on handling MOCOP. It develops a Hybrid Evolutionary Metaheuristics (HEMH) which incorporates both of DM-GRASP [18] and Path-relinking within the framework of the MOEA/D. The main goals are to capture the benefits of those techniques with providing cooperation, integration and adequate balance between intensification and diversification to improve the search capabilities. This can be achieved by applying Path-relinking or reproduction operators on high quality solutions obtained by DM-GRASP. The rest of the paper is organized as follows: section 2 presents some of the basic concepts and definitions. In section 3, an overview of GRASP and data mining is highlighted. The path-relinking strategy is discussed in section 4. Section 5 reviews the MOEA/D framework. The proposed HEMH is motivated and presented in section 6. In additions, experimental design and experimental results are involved in

sections 7 and 8 respectively. Finally, section 9 presents the conclusions and some directions for further research.

2. BASIC CONCEPTS AND DEFINITIONS

Without loss of generality, the MOOP can be formulated as:

$$Max\ F(x) = \big(f_1(x), f_2(x), \dots, f_m(x)\big) \tag{1}$$
$$s.t.: x \in \Omega.$$

Where, $F(x)$ is the m-dimensional objective vector, $f_i(x)$ is the ith objective to be maximized, $x = (x_1, \dots, x_n)^T$ is the n-dimensional decision vector and Ω is the feasible decision space. In case of $\Omega \subseteq \mathbb{Z}$, the MOOP is called *multiobjective combinatorial optimization problem* (MOCOP).

Definition 1: A solution x *dominates* y (noted as: $x \succcurlyeq y$) if: $f_i(x) \geq f_i(y)\ \forall i \in \{1, \dots, m\}$ and $f_i(x) > f_i(y)$ for at least one i.

Definition 2: A solution $x \in \Omega$ is called efficient (Pareto-optimal) if $\nexists y \in \Omega : y \succcurlyeq x$.

Definition 3: The *Pareto optimal set* (P^*) is the set of all efficient solutions: $P^* = \{x \in \Omega : \nexists y \in \Omega$ and $F(y) \geq F(x)\}$

Definition 4: The *Pareto front* (PF) is the image of the *Pareto optimal set* P^* in the objective space:

$$PF = \big\{F(x) = \big(f_1(x), f_2(x), \dots, f_m(x)\big) : x \in P^*\big\}$$

Definition 5: Given a reference point r^* and a weight vector $\Lambda = [\lambda_1, \dots, \lambda_m]$ such that $\lambda_i \geq 0, \forall i \in \{1, \dots, m\}, \sum_{i=1}^{m} \lambda_i = 1$,

The *weighted sum* (F^{ws}) and the *weighted Tchebycheff* (F^{Tc}) scalarizing functions corresponding to (1) are defined by (2) and (3) respectively as:

$$Max\ F^{ws}(x, \Lambda) = \sum_{i=1}^{m} \lambda_i\, f_i(x) \tag{2}$$
$$F^{Tc}(x, r^*, \Lambda) = Max_{1 \leq i \leq m}\big\{\lambda_i\big(r_i^* - f_i(x)\big)\big\} \tag{3}$$

Given a set of m knapsacks and a set of n items, the 0/1 Multiobjective Knapsack Problem (MOKSP) can be formulated as:

$$Max\ f_i(x) = \sum_{j=1}^{n} c_{ij}\, x_j, \forall i \in \{1, \dots, m\} \tag{4}$$
$$s.t.: \ \sum_{j=1}^{n} w_{ij}\, x_j \leq W_i, \forall i \in \{1, \dots, m\} \tag{5}$$
$$x = (x_1, \dots, x_n)^T \in \{0,1\}^n$$

Where, $c_{ij} \geq 0$ is the profit of the jth item in the ith knapsack, $w_{ij} \geq 0$ is the weight of the jth item in the ith knapsack and W_i is the capacity of the ith knapsack. When $x_j = 1$, it means that the jth item is selected and put in all knapsacks.

The MOKSP is NP-hard and can model a variety of applications. It was first formulated and solved by Zitzler & Thiele [22]. Since then, it has become a standard benchmark that has been solved by many other researchers [3] [21].

3. GRASP AND DATA MINING

3.1 GRASP Algorithm

GRASP [7] is a multi-start metaheuristics that has a *two phase* iterative process. In the first phase, the construction is invoked to build a completed solution. Then, the local search is applied on this solution to guarantee to be locally optimal in the second phase. This process is repeated until stopping criterion is met. The best solution found is taken as a result.

3.1.1 Construction

Greedy randomized construction adds randomize to greedy algorithm to produce a divers set of good quality starting solutions from which to start local search. It takes initially a partial solution S. Then, the *greedy function* g is evaluated for each unselected component in S. A restricted candidate list (*RCL*) is formed by the unselected components with $g \in [g_{min}, g_{min} + \alpha \times (g_{max} - g_{min})]$, where $\alpha \in [0,1]$ is a parameter to balance the greediness and randomness in S. A component is selected randomly from the *RCL* to be added to S. The whole process is repeated until S is completed.

3.1.2 Local Search

Local search is applied to improve the starting solutions produced by construction. Two basic strategies are often considered to accept local search moves, *first* and *best* improvements. In the *first* improvement, the first neighbor with better quality is accepted as a new current solution. In contrast, the *best* improvement examines all neighbors and accepts the best one as a new current solution. More sophisticated local search methods with good global search ability, such as simulated annealing and tabu search, have also been suggested to improve the starting solutions in GRASP [2].

3.2 DM-GRASP

In GRASP, iterations are performed independently from each other. Consequently, the knowledge acquired in the past iterations is not exploited in the subsequent iterations. The basic concept of incorporating data mining in GRASP is that patterns found in the high quality solutions obtained in earlier iterations can be used to improve the search process, leading to a more effective exploration of the search space, and consequently, a cooperative behavior is achieved instead of building each solution independently. The resulting heuristic is the DM-GRASP [18] that involves two phases [19]. The first one is to generate an elite set $\mathcal{D}$ through executing pure GRASP for n iterations and selecting the best solutions found. Then, data mining is applied on $\mathcal{D}$ to extract the set of patterns $\mathcal{P}$. Next, the hybrid phase is performed in which a number of slightly different iterations are executed. In these iterations, the construction receives a pattern $p \in \mathcal{P}$ as a partial solution from which a complete solution will be built.

4. PATH-RELINKING

Path-relinking was suggested to integrate intensification and diversification strategies in the context of TS and SS [6]. It generates new solutions by exploring trajectories that connect high quality solutions. Starting from the *starting solution* (x^s), path-relinking generates a path in the neighborhood space that leads toward the *guiding solution* (x^t). This can be accomplished through selecting moves that introduce attributes contained in x^t and incorporating them in an *intermediate solution* initially originated in x^s. It is observed that better solutions are found when the relinking procedure starts from the best of x^s and x^t. Because starting from the best gives the algorithm a better chance to investigate in more detail the neighborhood of the most promising solution [17]. Using path-relinking within GRASP as an intensification strategy applied to each locally optimal solution, was first proposed in [12]. It was followed by several extensions and applications [15] [16]. In HEMH, greedy randomized path-relinking [4] will be used as an intensification strategy to improve the performance and enhance the efficiency.

5. MOEA/D FRAMEWORK

MOEA/D [21] is a recently developed MOEA in which the decomposition idea is applied instead of dominance relation. The MOEA/D framework can be explained as a cellular MOEA [9] with a neighborhood structure in the m-dimensional weight space. A single cell with a single individual is located at the same place

as each weight vector in the *m*-dimensional weight space. That is, each cell has its own weight vector, which is used in the scalarizing function for evaluating the individual in that cell. Neighbors of a cell are defined by the Euclidean distance between cells in the weight space. The efficient solutions obtained over the search process are maintained in an external archive. To generate an offspring for a cell, two parents are randomly selected from its neighbors to apply reproduction. The offspring is compared with the individual in the current cell using the scalarizing function. If the offspring is better, it replaces the current individual. The offspring is also compared with each neighbor. The scalarizing function with the weight vector of each neighbor is used in the comparison. All neighbors, which are inferior to the offspring, are replaced with the offspring. This framework will be used by HEMH to carry out the proposed hybridization with DM-GRASP and greedy randomized path-relinking to enhance the performance and improve the search capabilities.

6. THE (HEMH) FOR 0/1 MOKSP

6.1 Motivations
This work can be motivated as follows:

- Using data mining to extract good patterns that help to build new solutions will achieve the cooperation among iterations.
- Applying reproduction on high quality solutions leads to produce high quality offspring.
- Incorporating path-relinking will help in discovering solutions beyond elite points as a post optimization strategy and will increase the intensification in these regions.
- Path-relinking gives the ability of investigating the non-convex regions and discovering the promising solutions lies on them.

6.2 The Proposed HEMH
Like MOEA/D [21], the HEMH needs a decomposition technique to convert the MOKSP in (4) into a set of single objective problems. The weighted sum approach described in (2) was used because it worked better than weighted Tchebycheff described in (3) on 0/1 MOKSP [8]. However, if we have a set of N uniformly distributed weight vectors $\{\Lambda^1, ..., \Lambda^N\}$, correspondingly we have N single-objective subproblems. HEMH attempts to simultaneously optimize these N subproblems. The set of neighbors of the ith subproblem includes all the subproblems with the T closest weight vectors $\{\Lambda^{i1}, ..., \Lambda^{iT}\}$ to Λ^i in terms of Euclidean distance. Each weight vector $\Lambda = [\lambda_1, ..., \lambda_m]$ can be generated according to (6). The number of different weight vectors that can be generated is defined by (7).

$$\sum_{i=1}^{m} \lambda_i = 1, \forall \lambda_i \in \{0/H, 1/H, ..., H/H\}, \forall i \in \{1, ..., m\} \quad (6)$$

$$N = \binom{H + m - 1}{m - 1}, \forall H \in \mathbb{Z}^+ \quad (7)$$

The HEMH framework is discussed with illustrations for its components and modules. Then, the whole procedure is explained.

6.2.1 HEMH framework
The HEMH framework contains two populations, main population and *Archive*. The main population consists of N members in which a solution is maintained for each search direction (subproblem). Each subproblem has T of neighbors. The *Archive* collects all efficient solutions explored over the search. It is periodically updated by new explored solutions. The search process consists of two basic phases, "*initialization*" and "*Main loop*". Initialization is responsible for obtaining an initial set of high quality solutions dispersed into PF. Whereas, the main loop

is responsible for discovering more new solutions in the most promising regions through applying greedy randomized path-relinking or reproduction on the set of high quality solutions previously obtained. Figure 1 clarifies the whole process.
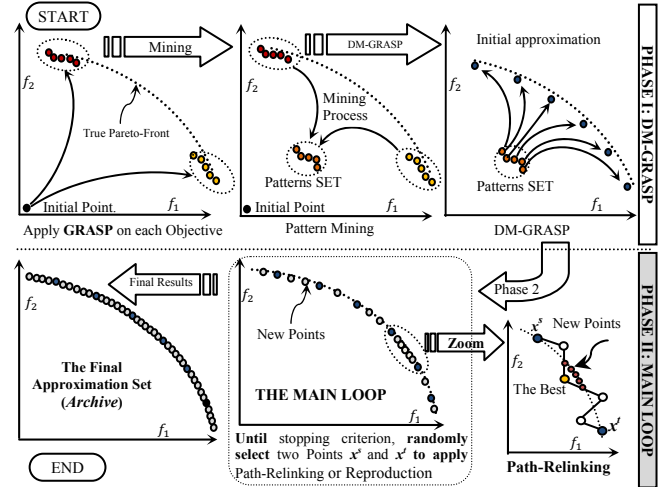

Figure 1: HEMH flow diagram

6.2.1.1 Initialization phase
DM-GRASP is applied to generate an initial set of high quality solutions to fill the main population. Firstly, original GRASP is applied on each objective function separately to construct a set of elite solutions from which a set of good patterns is extracted using data mining. Then, for each subproblem, one of the extracted patterns is selected as a partial solution to construct the current solution. DM-GRASP consists of Construction, Local search and Pattern-Mining modules. The procedures of both Construction and Local search were explained in [20]. The Pattern-Mining module receives as inputs the set *of minimum supports σ* that represent the minimum ratios of repetition of an item to be included in a pattern and the set of elite solutions *Archive*. It simply extracts the set of patterns $\mathcal{P}$ that achieve the *minimum supports σ* from *Archive*.

6.2.1.2 Main loop phase
In this phase, greedy randomize path-relinking or reproduction is applied on the solutions previously obtained in the initialization phase to intensify the search process in the regions surrounding the Pareto front. This means, concentrating the search efforts on the promising regions to discover new high quality solutions. Some of different modules used in this phase is explained.

6.2.1.2.1 Greedy Randomized Path Relinking
Greedy randomized path-relinking receives the inputs listed in Algorithm 1. Firstly, the best of x^s and x^t is chosen to start with. Then, the best fitness z^* and the best solution x^* are initialized. The candidate lists *CL* and *CL_comp* are constructed. Every unmatched j between x^s and x^t with $x_j^s = 0$ is inserted into *CL* in descending order according to the ratio in (8), whereas every unmatched j between x^s and x^t with $x_j^s = 1$ is inserted into *CL_comp* in increasing order according to (8).

$$\sum_{i=1}^{m} \lambda_i c_{ij} / \sum_{i=1}^{m} W_{ij} \quad (8)$$

The *RCL* is composed of the first $\alpha \times |CL|$ elements of *CL*. The procedure builds the path that connects x^s with x^t gradually by creating intermediate points through execution of the *Relinking loop*. Initially, the intermediate solution x is set to x^s. Then,

$\Delta(x, x^t)$ the number of unmatched between x and x^t is calculated. The next move is carried out by selecting one of unmatched ℓ^* to be matched, if the intermediate x is feasible, then, ℓ^* is randomly extracted from *RCL*, otherwise the first element of CL_{comp} is extracted to be ℓ^*. The new intermediate x is obtained by flipping the item (x_{ℓ^*}) corresponding to the selected index ℓ^* in the current intermediate x. If x is infeasible, the Greedy-Repair is invoked to get the feasible solution y. Then, the best fitness z^* and the best solution x^* are updated by y. This process is repeated until there is only one unmatched item between the current intermediate x and the *guiding* x^t. Finally, Local search is invoked to improve x^* only if $x^* \neq x^s$. Then, x^* is returned as an output.

Algorithm 1: GRPathRelinking ($x^s, x^t, \Lambda, \alpha, \beta$, *Archive*)
Inputs:
    x^s, x^t: *Starting & Guiding solutions.*
    $\Lambda = [\lambda_1, ..., \lambda_m]$: *Weighted vector of the current subproblem,*
    $\alpha \in [0,1] \subset \mathbb{R}$: *Controls greediness /randomness of move selection,*
Output: x^*: *the best solution found in the path from x^s to x^t;*
Begin:
01. **If** $F^{ws}(x^t, \Lambda) > F^{ws}(x^s, \Lambda)$ **then:** *//swapping to start with the best*
02. $x \leftarrow x^s;$ $x^s \leftarrow x^t;$ $x^t \leftarrow x;$
03. **End-If**
04. $x^* \leftarrow x^s;$ $z^* \leftarrow F^{ws}(x^s, \Lambda);$ *//initialize x^* and z^*.*
05. $CL \leftarrow \emptyset;$ $CL_{comp} \leftarrow \emptyset;$
06. **While** $\exists j : x_j^s \neq x_j^t \wedge x_j^s = 0$ with $Max_{j \in \{1,...,n\}} \frac{\sum_{i=1}^m \lambda_i c_{ij}}{\sum_{i=1}^m w_{ij}}$ **do:**
07. $CL \leftarrow CL \cup \{j\};$
08. **End-While**
09. **While** $\exists j : x_j^s \neq x_j^t \wedge x_j^s = 1$ with $Min_{j \in \{1,...,n\}} \frac{\sum_{i=1}^m \lambda_i c_{ij}}{\sum_{i=1}^m w_{ij}}$ **do:**
10. $CL_{comp} \leftarrow CL_{comp} \cup \{j\};$
11. **End-While**
12. $RCL \leftarrow$ The first $\alpha \times |CL|$ elements of *CL*. *//Define RCL*
13. $x \leftarrow x^s;$ $\Delta(x, x^t) \leftarrow \{j \in \{1, ..., n\} : x_j \neq x_j^t\};$ *//Ham. distance*
14. **While** ($|\Delta(x, x^t)| > 1$) **do:**   *//Relinking loop.*
15.   **If** x does not violate Equation (5) **then:**
16.   Randomly pick ℓ^* from *RCL*.
17. $CL \leftarrow CL \backslash \{\ell^*\};$
18.   **Else:**
19. $\ell^* \leftarrow$ The first element in CL_{comp}.
20. $CL_{comp} \leftarrow CL_{comp} \backslash \{\ell^*\};$
21.   **End-If**
22. $RCL \leftarrow$ The first $\alpha \times |CL|$ elements of *CL*. *//update RCL*
23. $x \leftarrow$ Flip $(x, \ell^*);$ *// flip ℓ^{*th} item in x.*
24.   **If** x violates Equation (5) **then:** $y \leftarrow$ **GreedyRepair** $(x, \Lambda);$
25.   **If** $(F^{ws}(y, \Lambda) > z^*)$ **then:**
26. $x^* \leftarrow y;$ $z^* \leftarrow F^{ws}(y, \Lambda);$ *//update x^* and z^**
27.   **End-If**
28. $\Delta(x, x^t) \leftarrow \{j \in \{1, ..., n\} : x_j \neq x_j^t\};$ *//update Ham. distance*
29. **End-While**
30. **If** $(x^* \neq x^s)$ **then:** $x^* \leftarrow$ **LocalSearch** $(x^*, \beta, \Lambda, Archive)$.
31. **return** x^*;

6.2.1.2.2 Greedy Repair

In **Greedy-Repair**, the infeasible solution x is repaired to be feasible. The main idea is to remove the items which have the minimum values of the ratio in (8) from the infeasible solution until becomes feasible. This ratio is calculated for each item in x based on a specified weight vector Λ that taken as a parameter.

6.2.1.2.3 Update-Solutions

The **Update-Solutions** procedure presented in algorithm 2 takes the solution y, t the number of solutions must be updated and *Pop* the range from which solutions are selected to be updated. The procedure starts with selecting a solution x^j from *Pop* for updating. Then, x^j is compared with y according the weighted

---

sum function (2) using the weight vector Λ^j, if y is better than x^j, then, x^j is replaced by y and deleted from *Pop*. This process is repeated until t iterations or *Pop* is empty.

Algorithm 2: Update-Solutions (y, t, Pop)
01. $c \leftarrow 0;$
02. **Repeat:**
03.   Randomly select an index j from *Pop*
04.   **If** $F^{ws}(y, \Lambda^j) \geq F^{ws}(x^j, \Lambda^j)$ **then:**
05. $x^j \leftarrow y;$ $c \leftarrow c + 1;$ *//update x^j by y and increment c.*
06. $Pop \leftarrow Pop \backslash \{j\};$
07.   **End-If**
08. **Until**($c = t$ or $Pop = \emptyset$);

6.2.2 The HEMH procedure

In algorithm 3, the proposed HEMH procedure which receives the input parameters listed below is described.

Algorithm 3: HEMH (*Stopping criterion*, $N, W_V, T, t, \delta, \alpha, \beta, \sigma, \varepsilon$)
Inputs:
    N: *Population Size or number of subproblems considered.*
    $W_V = \{\Lambda^1, ..., \Lambda^N\}$: *Set of N uniformly spread weight vectors.*
    T: *Size of neighborhoods of each subproblem.*
    $t \leq T$: *Maximum number of updated solutions.*
    $\delta \in [0,1]$: *Probability of selecting parents from the neighborhoods*
    $\alpha \in [0,1]$: *Parameter used at construction process.*
    $\beta \in [0,1]$: *Parameter used at the local search process.*
    σ: *Set of Minimum support for pattern-mining.*
    ε: *Minimum hamming distance allowed for applying path-relinking.*
Output: *Archive* : *all efficient solutions found over generations.*
Begin: *// initialization Phase*
01. **For** $i \in \{1, ..., N\}$ **do:** *//Define a set of T neighbors for each Λ^i*
02. $Neighbors^i \leftarrow \{i1, ..., iT\} : \Lambda^{i1}, ..., \Lambda^{iT}$ are the T closest to Λ^i
03. **End-For**
04. Let $\{\Lambda_{f_1}, ..., \Lambda_{f_m}\} \subseteq W_V$ be the set of all extreme weight vectors.
05. $Archive \leftarrow \emptyset;$
06. **For** $i \in \{1, ..., m\}$ **do:** *//Run GRASP for each objective separately*
07. $sol \leftarrow \emptyset;$
08. $sol \leftarrow$ **Construction** $(sol, \alpha, \Lambda_{f_i}, Archive);$
09. $sol \leftarrow$ **Local-Search** $(sol, \beta, \Lambda_{f_i}, Archive);$
10. **End-For**
11. $\mathcal{P} \leftarrow$ **PattarnMining**$(\sigma, Archive);$ *//construct the set of patterns*
12. **For** $i \in \{1, ..., N\}$ **do:** *//Initialize population using DM-GRASP*
13.   Randomly pick p from $\mathcal{P}$ *//choose a pattern*
14. $x^i \leftarrow$ **Construction** $(p, \alpha, \Lambda^i, Archive);$ *//Construct x^i using p.*
15. $x^i \leftarrow$ **Local-Search** $(x^i, \beta, \Lambda^i, Archive);$ *//Improve x^i.*
16. $FV^i \leftarrow F(x^i);$ *// Evaluation of x^i*
17. **End-For**
18. **While** *Stopping criterion* is not satisfied **do:** *//Main Loop Phase*
19.   **For** $i \in \{1, 2, ..., N\}$ **do:**
20.   Randomly generate $r \in [0,1];$
21.   **If** ($r < \delta$) **then:** *//Define Mating/updating rang*
22. $Pop \leftarrow Neighbors^i;$
23.   **Else:** $Pop \leftarrow \{1, ..., N\};$
24.   **End-If**
25.   Randomly pick j and k from *Pop* for recombination.
26.   **If**($\Delta(x^j, x^k) < \varepsilon$) **then:**
27. $y \leftarrow$ **Reproduction** $(x^j, x^k);$ *// Crossover & mutation*
28. $y \leftarrow$ **GreedyRepair** $(y, \Lambda^i);$ *// Repair if infeasible*
29.   **Else:**
30. $y \leftarrow$ **GRPathRelinking**$(x^j, x^k, \Lambda^i, \alpha, \beta, Archive);$
31.   **End-If**
32.   **Update-Solutions** $(y, t, Pop);$ *// Update Pop*
33. $Archive \leftarrow$ **Update-Archive** $(y, Archive);$
34.   **End-For**
35. **End-while**
36. **Return** $Archive$;
End

The procedure starts with identifying the set of neighborhoods for each subproblem i through calculating the Euclidian distance between Λ^i and each one of the set of all weight vectors $\{\Lambda^1, ..., \Lambda^N\}$ and choosing the T closest subproblems. The initial population members are initialized using DM-GRASP. Firstly, GRASP is applied on each objective function separately collecting elite solutions in the *Archive*. Then, pattern mining is applied on the *Archive* to extract the set of patterns $\mathcal{P}$. For each population member, a pattern $p \in \mathcal{P}$ is assigned as an input to the construction procedure to build a complete solution in which the Local search is applied on. The result is the ith member in the initial population. In the second phase, the search process is intensified in the promising regions beyond the solutions previously obtained through applying greedy randomized path-relinking or even reproduction (crossover and mutation). For each subproblem i, The Mating/Updating range (Pop) is determined to be either its neighborhood (*Local*) with probability equals to δ, or the whole population (G*lobal*). To generate the new offspring y, two parents x^j and x^k are randomly selected from Pop. Then, hamming distance $\Delta(x^j, x^k)$ is calculated. The greedy randomize path-relinking is applied to generate y only if $\Delta(x^j, x^k) \geq \varepsilon$. Otherwise, reproduction is considered. If y is infeasible, the Greedy-Repair module is invoked. Now, the offspring y must update both of the solution of the ith subproblem and t solutions from *Pop*. The Update-Solutions module is invoked to perform this task. The *Archive* is updated by every generated offspring. The whole process is repeated until stopping criterion is met. Finally, the *Archive* is returned as an output.

7. EXPERIMENTAL DESIGN
The HEMH is verified to approve its efficiency and effectiveness. The HEMH was implemented by C++. All experiments have been performed on HP Z600 workstation with (2 CPUs) Intel® X5670 2.93 GHz and 16.0 GB of RAM.

7.1 Tested Algorithms & Instances
To verify the performance of HEMH, some of the state-of-the art MOEAs are considered in this study such as NSGAII [3], SPEA2 [24], GRASPM [20] and MOEA/D [21]. The test instances listed below in table 1 are commonly used in the literature [21]. These instances are considered to perform the experimental design.

Table 1: Knapsack test instances

Instances			$N(H)$	HMEH $N(H)$	$MaxEvals$
Name	Knaps(m)	Items(n)			
KSP252	2	250	150(149)	75(74)	75000
KSP502	2	500	200(199)	100(99)	100000
KSP752	2	750	250(249)	125(124)	125000
KSP253	3	250	300(23)	153(16)	150000
KSP503	3	500	300(23)	153(16)	150000
KSP753	3	750	300(23)	153(16)	150000
KSP254	4	250	364(11)	165(8)	182000
KSP504	4	500	364(11)	165(8)	182000
KSP754	4	750	364(11)	165(8)	182000

7.2 Parameter settings
Here, the different parameters used for each MOEA is discussed. For MOEA/D, the parameter H which controls the number of weigh vectors or the population size (N), is determined with its corresponding (N) for each instance in table 1 above according to the complexity. Thus, all of MOEA/D, NSGA-II and SPEA2 use the same population size (N), whereas GRASPM uses (N) as the number of weight vectors. in HEMH, a small population size is

used to encourage path-relinking instead of reproduction. the values of (H) and their corresponding values of (N) used in HEMH for each instance is also listed in table 1. For NSGA-II, SPEA2 and MOEA/D, The initial population is generated randomly such that each member $x=(x_1, ..., x_n)^T \in \{0,1\}^T$, where $x_i=1$ with probability equals to 0.5. The maximum number of evaluations ($MaxEvals$) is used as stopping criterion for each MOEA. In both HEMH and GRASPM in which the local search is used, each fitness comparison performed inside the local search procedure is considered as an evaluation for fair comparison. For each compared MOEA, all efficient solutions observed over generations were collected in *Archive*. In these experiments, the same reproduction operator which combines the single-point crossover and the standard mutation was considered. Crossover was preformed with probability equals to 1, whereas mutation was performed for each item independently with probability equals to $1/n$. In both NSGAII and SPEA2, tournament selection is used with tournament size =2. The other control parameters are listed in table 2. Finally, the statistical analysis is applied on 30 independent runs for each MOEA on each test instance.

Table 2: Parameter used in MOEAs

Parameters	MOEAD	GRASPM	HEMH
Neighborhood in Mating: T	10	-	10
Max. no. of updated solutions: t	-	-	5
RCL definition ratio: α	-	0.1	0.1
Reconstruction ratio: β	-	0.5	0.5
Set of Minimum support: σ	-	-	{1}
Parents selection : δ	-	-	0.9
Minimal hamming distance: ε	-	-	10

7.3 Assessment Metrics
Let $A \subset \mathbb{R}^m$ and $B \subset \mathbb{R}^m$ be two approximations to the Pareto front (PF), $P^* \subset \mathbb{R}^m$ be a set of uniformly distributed points along PF (Reference Set) and $r^* \in \mathbb{R}^m$ be a reference point. The following indicators can be expressed as follows:

A) The Set Coverage ($\mathcal{C}$-metric) [22]**:**
This indicator is used to compare two approximation sets. The function $\mathcal{C}$ maps the ordered pair (A, B) to the interval $[0,1]$ as:

$$\mathcal{C}(A, B) = |\{u|u \in B, \exists v|v \in A : v \succcurlyeq u\}|/|B| \qquad (9)$$

Where, $\mathcal{C}(A, B)$ represents the percentage of the solutions in B that are dominated by at least one solution from A. $\mathcal{C}(B, A)$ is not necessarily equal to $1 - \mathcal{C}(A, B)$. In general, if $\mathcal{C}(A, B)$ is large and $\mathcal{C}(B, A)$ is small, then A is better than B in a sense.

B) The Hypervolume (s-metric) [22]**:**
The Hypervolume of a set A is defined as:

$$s(A, r^*) = \hbar(\cup_{u \in A}\{y|u \succcurlyeq y \succcurlyeq r^*\}) \qquad (10)$$

Where, $\hbar$ is the Lebesgue measure of a set. This indicator describes the size of the objective space that is dominated by points of A and dominates r^*. Here, r^* is chosen as the origin.

C) Generational and Inverted Generational Distance:
The Generational Distance (GD) & Inverted Generational Distance (IGD) of a set A are defined as:

$$GD(A, P^*) = \frac{1}{|A|}\sum_{u \in A}\{min_{v \in P^*} d(u, v)\} \qquad (11)$$

$$IGD(A, P^*) = \frac{1}{|P^*|}\sum_{u \in P^*}\{min_{v \in A} d(u, v)\} \qquad (12)$$

Where, $d(u, v)$ is the Euclidean distance between u and v in $\mathbb{R}^m$. The $GD(A, P^*)$ measures the average distance from A to the nearest solution in P^* that reflects the closeness of A to P^*. In contrast, the $IGD(A, P^*)$ measures the average distance from P^* to the nearest solution in A that reflects the spread of A to a certain

degree. The lower value of both $GD(A, P^*)$ and $IGD(A, P^*)$ means the better quality of A in terms of convergence and diversity respectively.

D) Maximum Spread (MS): [23]
This indicator evaluates the maximum extension covered by the nondominated solutions in A as follows:

$$MS(A) = \sqrt{\sum_{j=1}^{m}\left[\left(max_{i=1}^{|A|}f_m^i\right) - \left(min_{i=1}^{|A|}f_m^i\right)\right]^2} \quad (13)$$

Where, m is the number of objectives. One should note that the higher value indicates the better performance.

In these experiments, the reference set P^* is alternatively formed for each problem instance by gathering all nondominated solutions found by all of the compared MOEAs. Also, all approximation sets are normalized in the range [0, 1].

8. EXPERIMENTAL RESULTS
Here, the different simulation results are shown in details. Firstly, figure 2 shows the results of C-metric. It contains a chart (with scale 0 at the bottom and 1 at the top) for each ordered pair of the compared MOEAs. Each chart consists of nine box plots representing the distribution of C values. Each box plot (from left to right) represents an instance in table 1 (from top to down) respectively. A chart located in the raw of algorithm A and the column of algorithm B presents the values of coverage of approximations generated by algorithm B by approximations generated by algorithm A. It's clear from the results in figure 2 that HEMH and GRASPM outperform the rest MOEAs. It's also clear that HEMH performs better or even slightly better than GRASPM in all test instances.
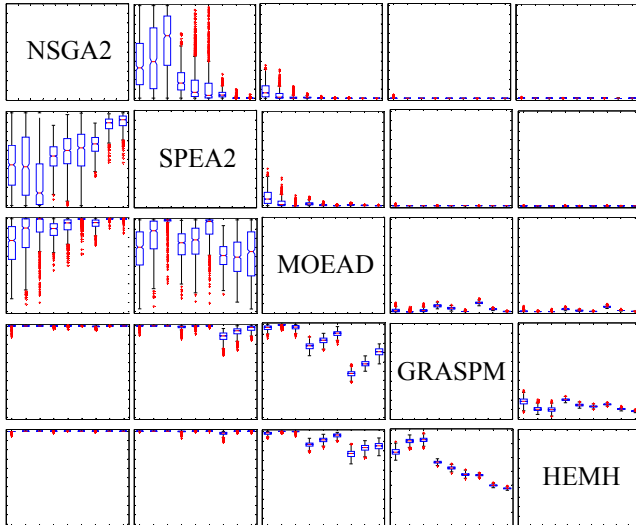

Figure 2: Comparisons of C-Metric.

According to the results of s-metric depicted by figure 3 and table 3 which contains the average values of the indicator achieved over 30 independent runs, it's clear that the HEMH outperforms all the compared MOEAs. Since, it has the maximum average s-metric values. This indicates the ability to improve both convergence and diversity. Whereas, GRASPM and MOEAD have the second and the third rank respectively in all test instances.

In table 4, the average values of the generational distance indicator are listed. Additionally, figure 4 visualizes these results. According to the generational distance results, the HEMH outperforms all the compared MOEAs. The GRASPM algorithm

achieves the second rank followed by the MOEAD which takes the third rank with respect to all test instances. This means that the HEMH has the capabilities of discovering solutions as near as possible to the Pareto front.

Table 3: The average hypervolume (s-metric)

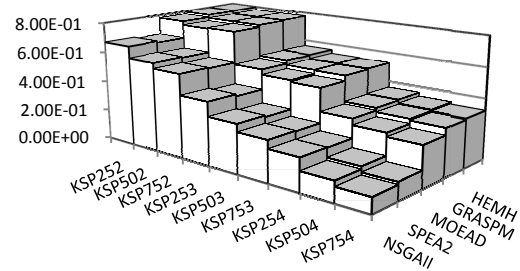| Instance	NSGAII	SPEA2	MOEAD	GRASPM	HEMH
KSP252	6.680E-01	6.576E-01	7.763E-01	7.948E-01	**7.976E-01**
KSP502	5.889E-01	5.842E-01	7.492E-01	7.710E-01	**7.757E-01**
KSP752	5.516E-01	5.469E-01	7.540E-01	7.702E-01	**7.751E-01**
KSP253	4.129E-01	3.994E-01	5.342E-01	5.538E-01	**5.580E-01**
KSP503	3.175E-01	3.070E-01	4.982E-01	5.247E-01	**5.308E-01**
KSP753	2.665E-01	2.599E-01	4.861E-01	5.211E-01	**5.270E-01**
KSP254	2.122E-01	2.094E-01	3.334E-01	3.502E-01	**3.553E-01**
KSP504	1.325E-01	1.498E-01	2.922E-01	3.235E-01	**3.306E-01**
KSP754	9.766E-02	1.145E-01	2.666E-01	3.124E-01	**3.216E-01**

Figure 3: Comparisons of s-Metric.

Table 4: The average generational distance (GD-metric)

| Instance	NSGAII	SPEA2	MOEAD	GRASPM	HEMH
KSP252	3.240E-03	3.142E-03	1.457E-03	4.020E-04	**2.307E-04**
KSP502	4.424E-03	4.555E-03	1.458E-03	3.500E-04	**1.747E-04**
KSP752	4.171E-03	4.993E-03	1.009E-03	2.889E-04	**1.462E-04**
KSP253	1.622E-03	1.377E-03	4.457E-04	1.771E-04	**1.261E-04**
KSP503	2.369E-03	1.984E-03	4.468E-04	1.312E-04	**9.126E-05**
KSP753	3.345E-03	2.912E-03	4.760E-04	1.041E-04	**7.739E-05**
KSP254	1.538E-03	1.042E-03	2.849E-04	1.516E-04	**1.140E-04**
KSP504	2.571E-03	1.576E-03	3.203E-04	9.534E-05	**8.983E-05**
KSP754	3.173E-03	2.017E-03	4.009E-04	8.047E-05	**6.864E-05**
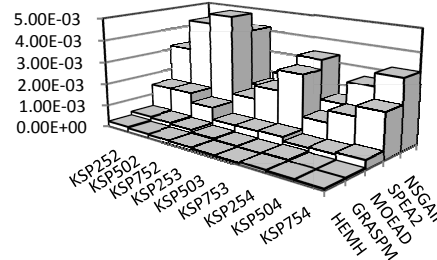
Figure 4: Comparisons of GD-Metric

The results of the inverted generational distance comparisons are listed in table 5 below, which contains the average values of the IGD-indicator over 30 independent runs. Also, figure 5 visualizes these results. It is clear that the HEMH outperforms all of the rest MOEAs, which reflects its ability to obtain solutions with good spread over the Pareto Frontier. The results also indicate that the GRASPM achieves the second rank followed by the MOEAD which take the third rank with respect to all test instances.

Table 6 and figure 6 below show the average values of the maximum spread indictor for all test instances. Based on these results, the HEMH has the superiority over other MOEAs,

followed by GRASPM. This assures their capabilities of exploring the extreme regions in the search space due to the local search used in both of them, which intensify the search on extremes.

**Table 5: The average inverted generational distance**

Instance	NSGAII	SPEA2	MOEAD	GRASPM	HEMH
KSP252	7.899E-03	8.608E-03	8.094E-04	3.468E-04	**3.161E-04**
KSP502	8.438E-03	8.595E-03	8.236E-04	2.467E-04	**1.717E-04**
KSP752	8.295E-03	8.126E-03	5.864E-04	2.055E-04	**1.378E-04**
KSP253	1.007E-03	1.153E-03	1.921E-04	9.910E-05	**8.606E-05**
KSP503	1.028E-03	1.143E-03	1.791E-04	9.015E-05	**7.263E-05**
KSP753	1.045E-03	1.124E-03	1.673E-04	8.099E-05	**6.300E-05**
KSP254	3.838E-04	4.127E-04	1.075E-04	8.232E-05	**7.264E-05**
KSP504	3.899E-04	3.968E-04	1.037E-04	7.686E-05	**6.203E-05**
KSP754	4.040E-04	4.081E-04	1.084E-04	7.057E-05	**5.611E-05**
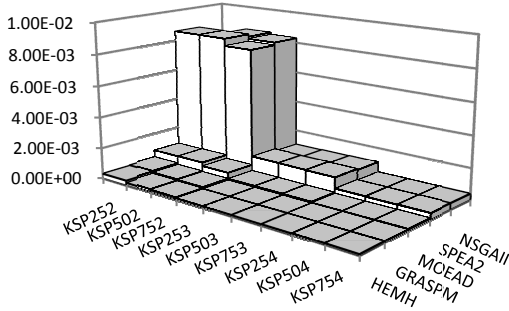
Figure 5: Comparisons of *IGD*-Metric

Table 6: The average maximum spread metric

Instance	NSGAII	SPEA2	MOEAD	GRASPM	HEMH
KSP252	5.168E-01	4.705E-01	1.373E+00	1.360E+00	**1.374E+00**
KSP502	3.788E-01	3.678E-01	1.309E+00	1.371E+00	**1.393E+00**
KSP752	2.598E-01	2.736E-01	1.317E+00	1.354E+00	**1.367E+00**
KSP253	8.916E-01	7.604E-01	1.650E+00	1.677E+00	**1.702E+00**
KSP503	6.653E-01	5.536E-01	1.653E+00	1.703E+00	**1.708E+00**
KSP753	4.758E-01	3.851E-01	1.644E+00	1.713E+00	**1.725E+00**
KSP254	1.234E+00	9.954E-01	1.903E+00	1.944E+00	**1.981E+00**
KSP504	1.066E+00	7.832E-01	1.902E+00	1.975E+00	**1.985E+00**
KSP754	8.273E-01	5.803E-01	1.838E+00	1.958E+00	**1.960E+00**
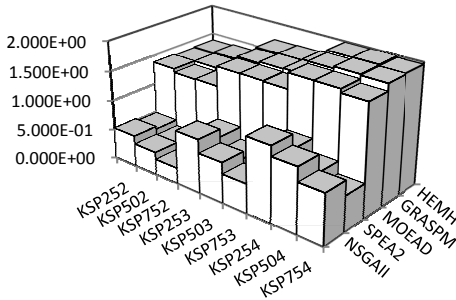
Figure 6: Comparisons of *MS*-Metric

Figure 7 involves 3 subfigures in which the approximation sets obtained by each MOEA are visualized for bi-objective instances KSP252, KSP502 and KSP752. Each subfigure contains 2 scatter graphs. The big one depicts the whole approximation sets whereas the small one in the left bottom corner focused on the part bounded by the small rectangle. In subfigure KSP252, HEMH and GRASPM achieves nearly the same points. From subfigures KSP502 and KSP752, it's clear that the solutions obtained by HEMH have the best quality. It is also noted from subfigures that

the quality of solutions obtained by HEMH is slightly increased gradually as the size of instance increased. This can be explained as, the larger the size of instance is, the more chance of hamming distance between any two selected solutions to increase. Consequently, path-relinking has more chance to be invoked instead of reproduction. This reflects the role played by path-relinking in improving the search capabilities of HEMH.
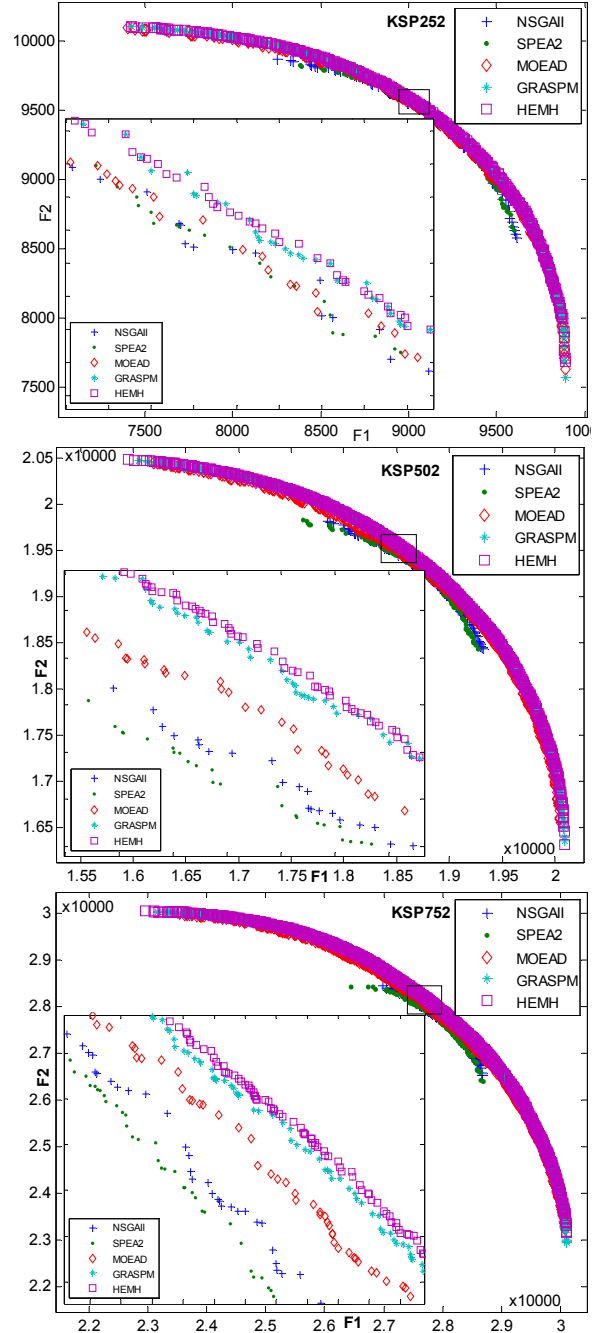
Figure 7: The achieved approximation sets of KSP252, KSP502 and KSP752 instances for each MOEA in 30 runs.

9. CONCLUSIONS

In this paper, a hybrid evolutionary metaheuristics (HEMH) based on DM-GRASP and greedy randomize path-relinking to solve

multiobjective knapsack problems was presented. The proposed HEMH was verified using a set of test instances commonly used in the literature. The HEMH was compared with four of the most popular MOEAs that considered as the state-of-the art. A set of quality assessment indicators was also considered to evaluate the performance for all the compared MOEAs. The experimental results indicate the superiority of the decomposition based MOEAs over the Pareto dominance based MOEAs. They also indicate the superiority of local search based MOEAs especially the HEMH. Since, it has an average performance highly competitive with respect to the compared MOEAs based on the assessment indicators used in the study. The main contribution of our algorithm is the combination among different metaheuristics techniques that intensify the search process in discovering the most promising regions in the search space and enhance the ability to explore good quality solutions. The second contribution is the ability to find a good approximation set of high quality solutions using a small set of uniformly distributed search directions due to the use of path-relinking and local search strategies. In the future work, the tuning parameters of the HEMH will be investigated as well as its convergence analysis. Additionally, the HEMH will be extended to handle other types of combinatorial optimization problems.

10. REFERENCES

[1] Blum, C., Roli, A. Metaheuristics in combinatorial optimization: Overview and conceptual comparison, *ACM Computing Surveys (CSUR), 35*, 3 (2003), 268-308.

[2] De la Pena, M. G. B. Heuristics and metaheuristics approaches used to solve the rural postman problem: A comparative case study. *In Proc. of the 4thInternational ICSC Symposium on Engineering of Intelligent Systems (EIS 2004)*

[3] Deb, K., Pratab, A., Agrawal S. and Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGAII. *IEEE Trans. on Evolutionary Computation*, 6, 2 (2002) 182-197.

[4] Faria, H., Jr. Binato, S. Resende, M.G.C. and Falcao, D.M. Power transmission network design by greedy randomized adaptive path relinking approach. IEEE Transactions on Power Systems, 20, 1 (2005) 43-49.

[5] Farina, M., Deb, K. and Amato, P. Dynamic multiobjective optimization problems: test cases, approximations, and applications. *IEEE Trans. on Evolutionary Computation, 8*, 5 (2004) 425-442.

[6] Glover, F. and Laguna, M. Fundamentals of scatter search and path relinking. *Control and Cybernetics 29, 3* (1999) 653-684.

[7] Glover, F., Kochenberger, G. *Handbook of Metaheuristics.* International Series in Operations Research & Management Science, Springer, 2003.

[8] Ishibuchi, H., Sakane, Y., Tsukamoto, N. and Nojima, Y. Adaptation of scalarizing functions in MOEA/D: An adaptive scalarizing function based multiobjective evolutionary algorithm. In *Proceedings of. 5th International Conference on Evolutionary Multi-Criterion Optimization,* (2009) 438-452.

[9] Ishibuchi, H., Sakane, Y., Tsukamoto, N. and Nojima, Y. Effects of using two neighborhood structures on the performance of cellular evolutionary algorithms for many-objective optimization. In *Proceedings of.2009 IEEE Congress on Evolutionary Computation,* 2009, 2508-2515.

[10] Jaszkiewicz, A. Do multiple objective metaheuristics deliver on their promises? A computational experiment on the set covering problem. *IEEE Trans. on Evolutionary Computation*, 7, 2(2003) 133-143.

[11] Jaszkiewicz, A. On the performance of multiple-objective genetic local search on the 0/1 knapsack problem - A comparative experiment. *IEEE Trans. on Evolutionary Computation*, 6, 4 (2002) 402-412.

[12] Laguna, M. and Marti R. GRASP and path relinking for 2-layer straight line crossing minimization. *INFORMS Journal on Computing, 11, 1* (1999) 44-52.

[13] Lozanoa, M. and García-Martínez, C. Hybrid metaheuristics with evolutionary algorithms specializing in intensification and diversification: Overview and progress report. *Computers &Operations Research* 37, 3 (2010) 481-497.

[14] Miettinen, K. *Nonlinear Multiobjective Optimization*. Boston, MAA: Kluwer, 1999.

[15] Resende, M. G. C. and Werneck, R. F. A hybrid multistart heuristic for the uncapacitated facility location problem. *European Journal of Operational Research*, 174, 1(Oct 2006) 54-68.

[16] Resende, M. G. C., Marti, R., Gallego, M. and Duarte, A. *GRASP and path relinking for the max-min diversity problem.* Technical report, AT&T Labs Research, Florham Park, NJ, 2008.

[17] Ribeiro, C., Uchoa, E. and Werneck, R. F. A hybrid GRASP with perturbations for the Steiner problem in graphs. *INFORMS Journal on Computing, 14, 3*(July 2002) 228-246.

[18] Ribeiro, M. H., Plastino, A. and Martins, S. L. Hybridization of GRASP metaheuristics with data mining techniques. *Mathematical Modeling and Algorithms*, 5 (2006) 23-41.

[19] Santos, L. F., Martins, S. L. & Plastino, A. Applications of the DM-GRASP heuristic: A survey. *International Trans. on Operational Research, 15, 4*(2008) 387-416.

[20] Vianna, D. S. and Claudio Arroyo, J. E. A GRASP Algorithm for the Multiobjective Knapsack Problem, In *Proceedings of XXIV International Conference of the Chilean Computer Science Society (SCCC'04)*, (2004) 69-75.

[21] Zhang, Q. and Li, H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Trans. on Evolutionary Computation*, 11, 6 (2007) 712-731.

[22] Zitzler, E. and Thiele, L. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto evolutionary algorithm. *IEEE Trans. on Evolutionary Computation*, 3 (1999) 257-271.

[23] Zitzler, E., Deb, K., and Thiele, L. comparison of multiobjective evolutionary algorithms: empirical results *.Evolutionary Computation* 8, 2(2000) 173-195.

[24] Zitzler, E., Laumanns, M. and Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization. *In Proc. of Evolutionary Methods for Design, Optimization and Control with Application to Industrial Problems (EUROGEN 2001).* Athena, Greece, 2001, 95-100.