A Hybrid Fine-Tuned Multi-Objective Memetic Algorithm

Xiuping GUO†a), Genke YANG†, Zhiming WU†, Nonmembers, and Zhonghua HUANG†, Student Member

SUMMARY In this paper, we propose a hybrid fine-tuned multi-objective memetic algorithm hybridizing different solution fitness evaluation methods for global exploitation and exploration. To search across all regions in objective space, the algorithm uses a widely diversified set of weights at each generation, and employs a simulated annealing to optimize each utility function. For broader exploration, a grid-based technique is adopted to discover the missing nondominated regions on existing tradeoff surface, and a Pareto-based local perturbation is performed to reproduce incrementing solutions trying to fill up the discontinuous areas. Additional advanced feature is that the procedure is made dynamic and adaptive to the online optimization conditions based on a function of improvement ratio to obtain better stability and convergence of the algorithm. Effectiveness of our approach is shown by applying it to multi-objective 0/1 knapsack problem (MOKP).

key words: hybrid, fine-tuned, memetic algorithm, multi-objective optimization, multi-objective 0/1 knapsack problem

1. Introduction

Many real-world optimization problems require taking into account multiple but often conflicting objectives. The difficulty in solving multi-objective problem (MOP) is not only due to its combinatorial complexity, but also to the research of all elements of the Pareto optimal set. A utility function with random weights often is used in aggregation methods (e.g., MOSA [1]) as a tool to guide search across all areas in feasible space. While in recent multi-objective evolutionary algorithms (MOEAs) (e.g., [2]–[6]), several concepts including crowding-distance, clustering, mating restriction and fitness sharing were introduced to find as many Pareto optimal solutions as possible, and also some form of elitism was used to improve the convergence speed to the Pareto front.

Presently, memetic algorithms or so called genetic local search algorithms incorporating local search into evolutionary algorithms have been applied to MOPs [7]–[11] and shown being promising methods for efficiently finding near Pareto-optimal solutions. The first multi-objective genetic local search (MOGLS) algorithm devised in [7] assigned fitness using a randomly selected utility function. Crossover and mutation were then performed for the parents drawn from the current approximation; a grid-based local perturbation is performed to reproduce incrementing solutions trying to fill up the discontinuous areas. Additional advanced feature is that the procedure is made dynamic and adaptive to the online optimization conditions based on a function of improvement ratio to obtain better stability and convergence of the algorithm. Effectiveness of our approach is shown by applying it to multi-objective 0/1 knapsack problem (MOKP).

The general MOP can be formulated as:

\[
\begin{align*}
\min & \quad f(x) = [f_1(x), f_2(x), \ldots, f_m(x)] \\
\text{subject to} & \quad g_i(x) \leq 0, \quad i = 1, 2, \ldots, n_g \\
& \quad h_j(x) = 0, \quad j = 1, 2, \ldots, n_h
\end{align*}
\]

where \(x\) is the decision vector, \(f(x)\) is the objective function, and \(g_i(x)\) and \(h_j(x)\) are the inequality and equality constraints, respectively. The goal is to find a set of solutions that are optimal with respect to all the objectives. The set of all such solutions is known as the Pareto front.

2. Multi-Objective Problem and Definitions

The general MOP can be formulated as:

\[
\begin{align*}
\min & \quad f(x) = [f_1(x), f_2(x), \ldots, f_m(x)] \\
\text{subject to} & \quad g_i(x) \leq 0, \quad i = 1, 2, \ldots, n_g \\
& \quad h_j(x) = 0, \quad j = 1, 2, \ldots, n_h
\end{align*}
\]
where $x$ is the decision vector, $X$ is the space of feasible solutions, and point $y$ is the objective vector.

For any two solutions $a, b \in X$, $a$ is said to dominate $b$ (also written as $a \succ b$) if
\[
\forall i = 1, 2, \ldots, n : f_i(a) \geq f_i(b) \quad \text{and} \quad \exists i : f_i(a) > f_i(b).
\]

A solution $x \in X$ is called Pareto optimal if there is no $x' \in X$ that dominates $x$. The image of the set of all Pareto optimal solutions is called Pareto front.

The weighted linear utility function used to evaluate the solution fitness in MOP is defined as:
\[
s(f, \lambda) = \sum_{i=1}^{n} \lambda_i f_i(x),
\]
where weight vector $\lambda = (\lambda_1, \lambda_2, \ldots, \lambda_n)$ meeting the condition:
\[
\forall i : \lambda_i \geq 0, \sum_{i=1}^{n} \lambda_i = 1.
\]

In Pareto-based fitness assignment method, a vector containing all objective values represents the fitness, and Pareto dominance relationships are used to determine the survival probability of candidate solutions.

3. The Proposed HFMOMA

3.1 HFMOMA Scheme

Figure 1 shows the feedback control scheme for HFMOMA, which receives initial parameters and outputs results. Each generation consists of two phases: local search first and recombination second. In the local search, for each random linear utility function is associated with an individual of the current population, whose utility function is maximized by a simulated annealing (SA). In the recombination, crossover operations are performed for the parents drawn at random from the union of the post-local search population $Pop$ and the external archive $EA$ used to store all found nondominated solutions during optimization, also, a grid-based local perturbation (GBLP) is used for further exploration.

Note that in Fig. 1, the procedure is adaptively tuned according to an online feedback improvement ratio $ir$, i.e., the moves number $iter$ on a temperature plateau in the SA, and the perturbations number $np$ in GBLP are controlled by the $ir$ along the evolution. The improvement ratio at generation $g$ is defined as:
\[
ir(g) = \frac{|\{ f(a) \in nd(g): \exists f(b) \in nd(g-1): f(a) > f(b) \}|}{\text{nd}(g)}.
\]
where $\text{nd}(g)$ and $\text{nd}(g-1)$ are respectively the sets of all nondominated solutions held in the $EA$ at generations $g$ and $(g-1)$. For normal convergence, as the generation proceeds, the improvement ratio decreases gradually towards a small value close to zero, indicating a high probability for further improvement of the evolution at the initial stage and a less probability to produce any new nondominated solutions that dominate the current ones at the final stage. Hence, the improvement ratio can be taken as a measure of optimization process and used to adjust the search for regulating the convergence speed at each generation. To filter out fluctuation due to any possible irregular $ir$ at each generation, the $ir$ at generation $g$ is stabilized in the study by taking the average $ir$ value of the last $l$ generations as defined by $\overline{ir}(g) = (1/l)\sum_{i=g-l+1}^{g} ir(i)$.

Based on the above description, the procedure for the HFMOMA is presented as below:

\begin{itemize}
  \item [Input:] $\text{popsize}$-population size, $\gamma$-cooling rate, $T_0$-initial temperature, $T_{\text{end}}$-end temperature, $\text{gen}$-the number of HFMOMA generations;
  \item [Output:] all generated nondominated solutions;
  \item [begin]
  \begin{itemize}
    \item Generate random initial population $Pop$;
    \item Set internal and external archives empty: $IA \leftarrow \emptyset$, $EA \leftarrow \emptyset$;
    \item Put all nondominated individuals of $Pop$ into $EA$;
    \item Set $g \leftarrow 1$;
    \item while $g \leq \text{gen}$ do
      \begin{itemize}
        \item Calculate $\overline{iter}(ir(g))$ and $np(ir(g))$ at current generation $g$;
        \item for each individual $x \in Pop$ do
          \begin{itemize}
            \item Draw at random a weight vector $\gamma$;
            \item Perform local search $SA(x, \gamma, \overline{iter}(ir(g)))$;
            \item Replace improved solution $x$ back into $Pop$;
          \end{itemize}
        \end{itemize}
      \end{itemize}
    \end{itemize}
  \item end
  \item [Procedure] $\text{HFMOMA}(\text{popsize}, \gamma, T_0, T_{\text{end}}, \text{gen})$
\end{itemize}

In this procedure, the internal archive $IA$ is functioned to hold the nondominated solutions generated during local search. Updating an archive with a solution involves in: 1) adding the solution to the archive if there is no other solution in the archive dominating the solution; 2) removing from the archive all solutions dominated by the solution. The union $EA \cup IA$ means that $EA$ is updated with each member of $IA$. The determinations of the dynamic parameters $\overline{iter}(ir(g))$ and $np(ir(g))$ will be described in Sects. 3.2 and 3.3. A weight vector $\lambda$ is generated in the following way:
\[
\lambda_i = \frac{\text{rand}_i}{(\text{rand}_1 + \text{rand}_2 + \text{rand}_n)},
\]
where $\text{rand}_i (i=1, 2, \ldots, n)$ is a non-negative real number.
3.2 Local Search with Simulated Annealing

The SA procedure in HFMOMA is depicted as follows:

\begin{algorithm}
\textbf{begin} \\
Set temperature $T \leftarrow T_0$ and set $k \leftarrow 0$; \\
Set IA as empty and place $x$ into IA; IA $\leftarrow \phi$, IA $\leftarrow x$; \\
while $T > T_{end}$ do \\
\textbf{while} $k < \text{iter}(ir(g))$ do \\
Construct randomly a feasible neighbor solution $x_1$ of $x$; \\
if $\Delta \leq \sum_{i=1}^{n} \alpha_i (f(x_i) - f(x'_i)) < 0$ then \\
Accept $x_1$ ($x \leftarrow x_1$) and update IA with $x_1$; \\
else if $\exp(-\Delta s(T) > \text{rand}(0,1))$ then \\
Accept $x_1$ ($x \leftarrow x_1$) \\
end \\
k++; \\
end \\
Update temperature $T \leftarrow \gamma T$ and set $k \leftarrow 0$; \\
end \\
\textbf{end}

\textbf{Procedure} SA($x, \gamma \text{ iter}(ir(g))$)
\end{algorithm}

In this process, any neighbor leading to an improvement of the utility function is accepted and used to update IA. Any deteriorating move is accepted with a probability $\exp(-\Delta s(T))$. The function $\text{rand}(0,1)$ generates a random number between 0 and 1.

Although a large number of moves on a temperature plateau in the SA often provide a fast convergence and broad local exploration at each generation, additional computational effort is required to perform these iterations. To solve the problem, the iterations number $\text{iter}$ on each temperature in the SA is made dynamic and adaptive to the optimization conditions based upon a function of the improvement ratio as defined in Sect. 3.1. There are two advantages for applying the improvement ratio to determine the number $\text{iter}$ of moves on a temperature plateau. First, $\text{iter}$ can be self-adjusted according to the online improvement ratio to regulate the evolving speed to the Pareto front. Normally, the higher the improvement ratio is, the easier the solutions are improved. Therefore, the iterations number $\text{iter}$ on each temperature in the SA should be set a low value with less computational effort if the improvement ratio is already above a satisfactory level. If the improvement ratio is below the satisfactory level, however, $\text{iter}$ may be increased so as to regenerate more “fitter” nondominated solutions for better improvement ratio. Second, the local fine-tuning is often more essential at the final stage than at the initial stage of the evolution in that computational stagnation often occurs approaching to the Pareto-optimal set. Thus, $\text{iter}$ is desired to follow the reverse trend of the improvement ratio along the evolution. As shown in Fig. 2, $\text{iter}$ is limited in the state “Good ir” with a high value of $ir$, and increased gradually to widen the search in the state “Medium ir” (i.e. as the optimization proceeds) until the evolution enters the state “Bad ir” with a low value of $ir$ denoting the approximation close to the final tradeoffs.

In view of these reasons, the $\text{iter}$ in the SA is set reversely to the improvement ratio of the evolution as illustrated in Fig. 2, and the $\text{iter}$ at generation $g$ is determined by the function of the $ir$ as formulated by:

$$\text{iter}(ir(g)) = \begin{cases} 
ub, & 0 \leq ir \leq \alpha_1; \\
ub - 2(\ub - \lb)(\frac{ir - \alpha_1}{\alpha_1})^2, & \alpha_1 < ir < \alpha_2; \\
\lb + 2(\ub - \lb)(\frac{\alpha_2 - ir}{\alpha_2 - \alpha_1})^2, & \alpha_2 \leq ir \leq 1.
\end{cases}$$

where $ir$ represents the improvement ratio at generation $g$, $\alpha_1$ and $\alpha_2$ are the fuzzy boundaries of different states, $\alpha$ is the satisfactory level for $ir$, $\lb$ and $\ub$ are respectively the lower bound and upper bound of $\text{iter}$.

3.3 Recombination with Pareto Dominance

In the recombination phase, a grid-based local perturbation GBLP is carried out after the crossover operation for broader exploration. As illustrated in Fig. 3, GBLP works by dividing $n$ dimensional objective space occupied by all members of the external archive into $G_1 \times G_2 \times \cdots \times G_n$ grids,
where $G_k, k=1,2,...,n$, represents the grids number in the $k$th objective dimension (e.g. in Fig. 3, $G_1=G_2=4$). Thus, the grid width in the $k$th objective dimension, $w_k$, is calculated as:

$$w_k = \frac{F_{\text{max}} - F_{\text{min}}}{G_k}, k = 1,2,...,n,$$

where $F_{\text{min}}^k$ and $F_{\text{max}}^k$ are the lower boundary and upper boundary of the occupied region in the $k$th objective dimension (e.g., in Fig. 3, $F_{\text{min}}^1$ and $F_{\text{max}}^1$).

Each individual in the grids is located in a particular grid region. The location of an individual $x$ in the $k$th objective dimension, $l_k$, is calculated as [13]:

$$l_k = \text{mod}(d_k, w_k) + 1, k = 1,2,...,n,$$

where $d_k=f_k - F_{\text{min}}^k$ is a distance measure, $f_k$ is the $k$th objective of the solution $x$, and the function mod($a$, $b$) denotes the modulus (integer part) after division $a/b$.

The density of a grid is defined as the number of individuals dwelling in it and sharing the same $l_k$, $k=1,2,...,n$. In GBLP, the parents selected by a tournament are perturbed to produce incrementing individuals to fill up the gaps on existing tradeoff surface. The tournament selection is solely based on the density of an individual’s location, i.e., as demonstrated in Fig. 3, the parents located in the less crowded regions have higher probability to be perturbed as compared to those located in the more crowded regions.

Each selected parent in GBLP is perturbed to produce a number of offspring using an extension of mutation, in which more than one locally perturbed child are formed per parent. Consider an individual being coded in an $m$-digit chromosome $x=(x_1,x_2,...,x_n)$, where $s$ is coding index such that $s=1$ represents the most significant index and $s=m$ denotes the least significant index of the chromosome. As shown in Fig. 3, in order to increase the perturbation probability within the parent’s neighborhood rather than outside the neighborhood, the perturbation probability $p_s$ for the gene $x_s$ of $x$ is determined by a sigmoid function [14], which makes the more significant a digit in $x$ is, the lower its perturbation probability in GBLP is.

Concerning the same reasons as described in Sect. 3.2 for determining the iter in the SA, to reduce the computational effort and regulate the convergence speed, the number $np$ of perturbations per parent in GBLP also is fine-tuned based on the improvement ratio, following the same trend as shown in Fig. 2. For simplicity, the $np$ at generation $g$ is calculated as:

$$np(ir(g)) = \begin{cases} ubnp, & 0 \leq ir \leq a_1; \\ ubnp - 2(ubnp - lbnp)(\frac{ir-a_1}{g})^2, & a_1 < ir < a_2; \\ lbnp + 2(ubnp - lbnp)(\frac{ir-a_2}{g})^2, & a_2 \leq ir \leq 1, \\ lbnp, & \end{cases}$$

where $lbnp$ and $ubnp$ are the lower bound and upper bound of $np$ respectively.

Thus, the recombination procedure is given as:

begin
  Set $recb=0$;
  while $recb < recbMax$ do
    Draw randomly two solutions $x_2$ and $x_1$ from $EA \cup Pop$;
    Perform crossover for $x_2$ and $x_1$ obtaining $x_3$;
    if $x_3$ is not dominated by $EA$ then
      Update $EA$ with $x_3$;
    end
    $recb++$;
  end
  Set intermediate population empty: $Pop' \leftarrow \phi, r=0$;
  while $r<\text{popsize}$ do
    Draw a parent $y$ by tournament from $EA$;
    Perform GBLP with $np(ir(g))$ number of perturbations for $y$, updating $EA$ with any perturbed child not dominated by $EA$ and returning a set $pc$ not dominated by $EA$;
    if $pc \neq \phi$ then
      Use tournament to select a child in $EA$;
    else
      Select the child $y_{pc}$ with the lowest grid density;
    end
    Place offspring $e$ into intermediate population $Pop'$,
    $r++$;
  end
  Update population: $Pop \leftarrow Pop'$
end

Procedure Recombination($Pop$, $EA$, $np(ir(g))$)

In this procedure, a threshold number $recbMax$ of crossovers are performed on the solutions randomly chosen from the union of $Pop$ and $EA$. Then GBLP is used to perturb each parent selected from $EA$, obtaining a set $pc$ of the perturbed children not dominated by $EA$. If $pc$ is not empty, the offspring with the lowest location density is placed into the intermediate population $Pop'$, otherwise, the tournament selection is used to select a new offspring from $EA$ to join $Pop'$.

4. Numerical Experiments

We use the MOKP described by Zitzler and Thiele [6] to test the performance of HFMOMA. The approximations generated by our method were compared to the results found by SPEA, SPEA2, NSGA-II, MOGLS [10] and M-PAES. The instances and the approximations generated by SPEA, SPEA2 and NSGA-II are available on the web-site [15]. The results of SPEA2 and NSGA-II are available only for instances with 750 items. The results of MOGLS [10] to the same set of instances are available on the web-site [16] (MOGLS50 is taken here in that it achieves the best results). The results of M-PAES on the same set of instances were compared to the results found by Celeron(R) CPU 2.40 GHz and 512 M memory.

4.1 Multi-Objective 0/1 Knapsack Problem

The MOKP is defined in the following way [6]. Given $n$ knapsacks and $m$ items, with $p_{i,j}$ is the profit of item $j$ according to knapsack $i$, $w_{i,j}$ is the weight of item $j$ according
to knapsack \( i \), \( c_i \) is the capacity of knapsack \( i \), \( p_{i,j}, w_{i,j} \) and \( c_i \) are all positive constants. A vector \( \mathbf{x}=(x_1,x_2,\ldots,x_m)\in[0,1]^m \), is searched to maximize:

\[
\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))
\]

s.t. \( \sum_{j=1}^m w_{i,j} x_j \leq c_i \), \( i = 1, 2, \ldots, n \).

where \( x_j = 1 \) iff item \( j \) is selected.

The same set of nine MOKP instances as used in [6] is tested in the experiment. The instances have 2, 3 and 4 objectives and 250, 500 and 750 items, denoted as 2-250, 3-250, 4-250, etc. The instances were generated randomly with uncorrelated profits and weights. The capacities of knapsacks are equal to half the total weight of items.

The HFMOMA is adapted to the MOKP on the basis of the work of Zitzler and Thiele [6], where the solution is encoded as a binary string \( \mathbf{x}\in\{0,1\}^m \), and a greedy repair algorithm is proposed for the problem. The algorithm removes items from an unfeasible solution step by step. The items are removed in the increasing order of the maximum profit/weight ratio per item; for item \( j \) the maximum profit/weight ratio \( q_j \) is given by the equation:

\[
q_j = \max_{i=1}^n \left( \frac{p_{i,j}}{w_{i,j}} \right), \quad j = 1, 2, \ldots, m.
\]

Thus, we sort the items decreasingly according to the \( q_j \), i.e., those achieving the lowest profit per weight unit are removed first. In GBLP, the larger the \( q_j \) is, the lower the perturbation probability for the item \( j \) is.

### 4.2 Performance Metrics

One of the performance metrics used here is the coverage of two sets to the Pareto front. Let \( A \) and \( B \) be two approximations. The coverage \( \tilde{C} \) [17] is defined as:

\[
\tilde{C}(A, B) = \frac{|\{f(b') \in B; \exists f(a') \in A : f(a') > f(b')\}|}{|B|}.
\]

The value \( \tilde{C}=1 \) means that all points in \( B \) are dominated by some points in \( A \). The value \( \tilde{C}=0 \) means none of the points in \( B \) are dominated by any of the points in \( A \). Generally, \( \tilde{C}(A, B) \neq \tilde{C}(B, A) \).

As the second performance metric, we use the front spread \( (FS) \) [18] to measure the size of the objective space covered by an approximation. A larger \( FS \) value is preferable. Let approximation \( A \) be the image of solutions set \( \mathcal{S} \) for \( n \)-objective problem. The \( FS \) of \( A \) is defined as:

\[
FS(A) = \sqrt{\sum_{i=1}^n \max_{\mathbf{x} \in \mathcal{S}} \left( f_i(\mathbf{x}^0) - f_i(\mathbf{x}^1) \right)^2}.
\]

where \( f_i(\mathbf{x}^0) \) and \( f_i(\mathbf{x}^1) \) are respectively the \( i \)th objective values of solutions \( \mathbf{x}^0 \) and \( \mathbf{x}^1 \).

### 4.3 Results Comparison

The HFMOMA was run 30 times with different random seeds on each instance, for ten generations. An intensive experiment was performed to find good parameter settings for the HFMOMA. In Table 1 are described, for each instance, population size \( \text{popsize} \), lower bound \( lb \) and upper bound \( ub \) of the \( \text{iter} \) in the SA, grids number \( G_i \) in the \( i \)th objective dimension and the crossovers number \( \text{recbMax} \). The other parameters for each instance were set as: \( \alpha_1=0.001, \alpha_2=0.9, \alpha_5=0.5, l=2, \text{lbnp}=10, \text{ubnp}=50 \), the lower bound and upper bound of \( p_i \) in GBLP are set as 0.03 and 0.8 respectively. The parameters in the local searcher \( \text{SA} \) for each instance were set as: \( T_0=0.0001, T_{\text{cool}}=0.000001 \) and \( \gamma=0.95 \).

The mutation rate in the HFMOMA was set as \( 4/L \) (where \( L \) is the bits number in the chromosome) as in M-PAES [11] for each instance, but was set as \( 1/L \) in GBLP.

4.3 Results Comparison

Table 1 Parameter settings used in HFMOMA for each test instance.

<table>
<thead>
<tr>
<th>Instance</th>
<th>( \text{popsize} )</th>
<th>( lb )</th>
<th>( ub )</th>
<th>( G_i )</th>
<th>( G_j )</th>
<th>( G_k )</th>
<th>( G_l )</th>
<th>( \text{recbMax} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-250</td>
<td>10</td>
<td>500</td>
<td>700</td>
<td>8</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>100</td>
</tr>
<tr>
<td>2-500</td>
<td>10</td>
<td>700</td>
<td>900</td>
<td>8</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>100</td>
</tr>
<tr>
<td>2-750</td>
<td>10</td>
<td>900</td>
<td>1100</td>
<td>8</td>
<td>8</td>
<td>-</td>
<td>-</td>
<td>100</td>
</tr>
<tr>
<td>3-250</td>
<td>15</td>
<td>800</td>
<td>1000</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>-</td>
<td>75</td>
</tr>
<tr>
<td>3-500</td>
<td>15</td>
<td>1000</td>
<td>1200</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>-</td>
<td>50</td>
</tr>
<tr>
<td>3-750</td>
<td>15</td>
<td>1000</td>
<td>1200</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>-</td>
<td>50</td>
</tr>
<tr>
<td>4-250</td>
<td>15</td>
<td>1000</td>
<td>1200</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>50</td>
</tr>
<tr>
<td>4-500</td>
<td>20</td>
<td>1000</td>
<td>1200</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>50</td>
</tr>
<tr>
<td>4-750</td>
<td>15</td>
<td>1000</td>
<td>1200</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>32</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 2 Average execution time consumed by HFMOMA and M-PAES for each test instance.

<table>
<thead>
<tr>
<th>Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>

4.3 Results Comparison

The HFMOMA was run 30 times with different random seeds on each instance, for ten generations. An intensive experiment was performed to find good parameter settings for the HFMOMA. In Table 1 are described, for each instance, population size \( \text{popsize} \), lower bound \( lb \) and upper bound \( ub \) of the \( \text{iter} \) in the SA, grids number \( G_i \) in the \( i \)th objective dimension and the crossovers number \( \text{recbMax} \). The other parameters for each instance were set as: \( \alpha_1=0.001, \alpha_2=0.9, \alpha_5=0.5, l=2, \text{lbnp}=10, \text{ubnp}=50 \), the lower bound and upper bound of \( p_i \) in GBLP are set as 0.03 and 0.8 respectively. The parameters in the local searcher \( \text{SA} \) for each instance were set as: \( T_0=0.0001, T_{\text{cool}}=0.000001 \) and \( \gamma=0.95 \).

The mutation rate in the HFMOMA was set as \( 4/L \) (where \( L \) is the bits number in the chromosome) as in M-PAES [11] for each instance, but was set as \( 1/L \) in GBLP.

The comparison results based on the coverage metric are presented in Figs. 4 and 5. A simple version of box plots [12] is used to visualize the distribution of \( \tilde{C} \) samples. The thick line represents the median value, the upper and lower ends of the box are the upper and lower quartiles, and the ends of the vertical line are the minimum and maximum values.

From Fig. 4, we can see that HFMOMA obviously outperforms other given methods concerning the coverage metric. For each instance, the HFMOMA approximations dominate a large percentage of the points generated by MOGLS and M-PAES, while the MOGLS and M-PAES fronts dominate a rather small portion of the points produced by HFMOMA, especially for instances with four objectives. Besides, comparing HFMOMA and SPEA, the HFMOMA approximations dominate almost 100% of the SPEA fronts for all instances except for instance 2-250.
Fig. 4 Box plots based on the $\tilde{C}$ metric of nine instances. In each chart, the three box plots to the left associate with 2 knapsacks and (from left to right) 250, 500 and 750 items. Correspondingly, the three middle box plots associate with 3 knapsacks and three to the right with 4 knapsacks.

Fig. 5 Box plots based on the $\tilde{C}$ metric of three instances with 750 items. The three box plots associated with 2, 3 and 4 knapsacks (from left to right).

Fig. 6 Approximations obtained by particular algorithms.

Compared with SPEA2 and NSGA-II for instances with 750 items, Fig. 5 shows that for instances 3-750 and 4-750, the coverage metric indicates clear advantage of HFMOMA over SPEA2 and NSGA-II, whose fronts are dominated by the HFMOMA fronts by nearly 100%.

Figures 6(a) and (b) present the fronts generated by
HFOMA, MOGLS, SPEA and M-PAES for instances 2-500 and 2-750 respectively. Figure 6(c) shows the points found by HFOMA, SPEA2 and NSGA-II for instance 2-750. Clearly, the fronts obtained by HFOMA are closer to the Pareto front for all instances. Results for instance 2-250 are not shown in Fig. 6 because the sets of points overlap and the figure is hardly perceptible.

The results concerning the front spread are reported in Tables 3 and 4, which present the average $FS$ values for the approximations generated by the given algorithms. Table 3 shows that HFOMA obtains more diversified solutions compared to M-PAES and SPEA for all instances in that it achieves larger $FS$ values, but as is not the case when comparing with MOGLS [10] in terms of the $FS$ metric. For instances with 750 items, Table 4 gives clear evidence that the proposed method outperforms SPEA2 and NSGA-II according to the front spread values.

### Table 3 Comparison of MOGLS, M-PAES, SPEA and HFOMA: Front Spread ($FS$).

<table>
<thead>
<tr>
<th>Instance</th>
<th>MOGLS</th>
<th>M-PAES</th>
<th>SPEA</th>
<th>HFOMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-250</td>
<td>3465</td>
<td>2577</td>
<td>1422</td>
<td>3250</td>
</tr>
<tr>
<td>2-500</td>
<td>5346</td>
<td>3868</td>
<td>1607</td>
<td>4844</td>
</tr>
<tr>
<td>2-750</td>
<td>8935</td>
<td>6514</td>
<td>3306</td>
<td>8300</td>
</tr>
<tr>
<td>3-250</td>
<td>4437</td>
<td>2998</td>
<td>2062</td>
<td>4283</td>
</tr>
<tr>
<td>3-500</td>
<td>7763</td>
<td>3565</td>
<td>3547</td>
<td>7629</td>
</tr>
<tr>
<td>3-750</td>
<td>11260</td>
<td>9490</td>
<td>3858</td>
<td>10511</td>
</tr>
<tr>
<td>4-250</td>
<td>4857</td>
<td>2593</td>
<td>2643</td>
<td>4342</td>
</tr>
<tr>
<td>4-500</td>
<td>8282</td>
<td>4391</td>
<td>4452</td>
<td>8693</td>
</tr>
<tr>
<td>4-750</td>
<td>13498</td>
<td>4617</td>
<td>5508</td>
<td>12256</td>
</tr>
</tbody>
</table>

### Table 4 Comparison of SPEA2, NSGA-II and HFOMA: Front Spread ($FS$).

<table>
<thead>
<tr>
<th>Instance</th>
<th>SPEA2</th>
<th>NSGA-II</th>
<th>HFOMA</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-750</td>
<td>4085</td>
<td>3416</td>
<td>8300</td>
</tr>
<tr>
<td>3-750</td>
<td>6220</td>
<td>6582</td>
<td>10511</td>
</tr>
<tr>
<td>4-750</td>
<td>8269</td>
<td>11285</td>
<td>12256</td>
</tr>
</tbody>
</table>

5. Conclusions

A hybrid fine-tuned multi-objective memetic algorithm (HFOMA) is proposed. To achieve global exploitation and exploration, the aggregation functions are hybridized with the Pareto dominance relation to perform the fitness evaluation. The algorithm is implemented by using a simulated annealing combined with the use of a population, crossover and a grid-based local perturbation. To obtain better convergence of the algorithm, the procedure is fine-tuned based on the online feedback improvement ratio.

To illustrate the effectiveness of the HFOMA, we compared it with several recent algorithms on the multi-objective 0/1 knapsack problem. The computational results reveal that, for all tested instances, the proposed algorithm is completely superior to M-PAES, SPEA, SPEA2, NSGA-II on the metrics of the coverage and the front spread. Although HFOMA is a little inferior to MOGLS based on the front spread in most instances, it obviously outperforms MOGLS in the sense of Pareto optimality.

### Acknowledgements

The authors gratefully acknowledge the support from the National Science Foundation of China (No. 60174009).

### References

Xiuping Guo received the M.E. degree in Automatic Control Engineering from Kunming University of Science and Technology, China, in 2003. She is currently completing her Ph.D. in Computer Integrated Manufacturing System Laboratory, Shanghai Jiaotong University, China. Her research interests include scheduling, Logistics and Applications of Evolutionary Algorithms.

Genke Yang is a professor in the Department of Automation, Shanghai Jiaotong University, China. He received his Ph.D. from Xi’an Jiaotong University, China. His research interests include Methods of System Optimization, Scheduling, Logistics and Hybrid System.

Zhiming Wu is a professor in the Department of Automation, Shanghai Jiaotong University, China. He is an editor for 4 academic journals (in Chinese): Acta Automation, Control Theory and Application, Control and Decision, and Journal of Shanghai Jiaotong University. His research interests include Discrete Event Systems, Hybrid Dynamic Systems and their Industrial Application.

Zhonghua Huang is currently a Ph.D. student in Shanghai Jiaotong University, China. He received M.E. degrees in Nanjing University of Science and Technology, China, in 2002. His research interests include Petri nets theory and applications, especially the scheduling and control for automated manufacturing systems.