On Solving Complex Optimization Problems with Objective Decomposition

Yiu-ming Cheung
Department of Computer Science
Hong Kong Baptist University
Kowloon Tong, Hong Kong
ymc@comp.hkbu.edu.hk

Fangqing Gu
Department of Computer Science
Hong Kong Baptist University
Kowloon Tong, Hong Kong
fqgu@comp.hkbu.edu.hk

Abstract

This paper addresses the complex optimization problem, of which the objective function consists of two parts: One part is differentiable and unimodal, meanwhile the other part is non-differentiable and multi-modal. Accordingly, we decompose the original objective function into several relatively simple sub-objective ones, which therefore formulate as a multiobjective optimization problem (MOP). To solve this MOP, we propose a simulated water-stream algorithm (SWA) inspired by the natural phenomenon of water streams. The water streams with a hybrid process of downstream and penetration towards the basin are analogous to the process of finding the minimum solution in an optimization problem. The SWA featuring a combination of deterministic search and heuristic search generally converges much faster than the existing counterparts with a considerable accuracy enhancement. Experimental results show the efficacy of the proposed algorithm.

1 Introduction

Optimization problems are quite common in a variety of scientific areas. In the last decades, a number of optimization algorithms based on different theories and methodologies have been presented, which can be roughly summarized into two categories. The first category is the classical optimization algorithms, e.g. Conjugate Gradient Method [1], and Quasi-Newton Method [2]. In general, supposing the objective function is differentiable, these algorithms are able to converge to a global optimal solution quickly for a convex optimization problem [3], but may almost always lead to a local optimal solution if the objective function is multi-modal. Recently, some optimization algorithms, e.g. tunneling algorithm [4] and filled function method [5], have been developed for multi-modal optimization problems. Numerical studies have shown their effectiveness in their application domain, but their computation is generally laborious without guaranteeing to converge to a global optimal solution in the high-dimensional optimization problems.

The other category is heuristic algorithms, e.g. Evolutionary Algorithm [6, 7, 8, 9], Particle Swarm Optimization [10, 11], and Ant Colony Algorithm [12], all of which have achieved promising results in dealing with complex problems. Compared to the methods in the first category, these heuristic algorithms tackle the problem as a “black” box, i.e., without any hypotheses on the objective functions in terms of modality and differentiability. Nevertheless, they do not make good use of a priori knowledge of the problem. In general, the convergence speed of these algorithms is much slower in comparison with the methods in the first category.

In this paper, we will address a kind of the complex optimization problem, of which the objective function is non-differentiable, or even multi-modal as a whole. Nevertheless, part of such an objective function is differentiable, or even is convex. To solve this kind of optimization problems, the first category optimization algorithms stated above may be invalid or converge to a local optimal solution, whereas the second category algorithms do not make the best use of the property of the problems. To the best of our knowledge, there has been relatively little research conducted on these problems. In this paper, we firstly decompose the objective function of the original complex problem into several relatively simple sub-objective ones, which therefore formulate as a multiobjective optimization problem (MOP). To solve this MOP, we will present an effective algorithm, namely the simulated water-stream algorithm (SWA), which is inspired by the natural phenomenon of water streams. The water streams with a hybrid process of downstream and penetration towards the basin is analogous to the process of finding the minimum solution in an optimization problem. Specifically, a deterministic search is utilized to simulate the process of downstream, while two heuristic searches are introduced to simulate the necessity and contingency of the water penetration. The necessity means that most of the wa-
eter streams will penetrate downward. A neighborhood is defined for each water stream. The water stream penetrates to the lowest location found by the neighbor ones by a heuristic search, and the speed of the penetration is adaptively controlled by the speed of downstream. The contingency implies that a small portion of water streams may penetrate laterally, even upward. It is represented by a heuristic search based on random perturbation. As a result, the SWA featuring a combination of the deterministic search and heuristic search inherits the merits of those methods in the above-stated two categories. Numerical studies have shown the promising results of SWA in comparison with the existing counterparts.

The remainder of this paper is organized as follows: Section II describes the objective decomposition technique on the optimization problems. Section III gives a detailed description of the SWA. In Section IV, we show the experimental results to compare the SWA with the existing counterparts. Finally, we draw a conclusion in Section V.

2 Collaborative Alternately Optimal each Sub-Objective

2.1 Objective Decomposition

Without loss of generality, an optimization problem can be formulated as follow:

$$\min_{x \in D} F(x) \tag{1}$$

where $D \subset R^n$ is the decision space and $x \in D$ is the decision variable. In many cases, the objective $F(x)$ is non-differentiable or multi-modal, but part of which may be differentiable. That is, it can be formulated as the sum of some relatively simple functions:

$$F(x) = f_1(x) + \cdots + f_m(x) \tag{2}$$

where $f_1(x), \cdots, f_m(x)$ are called the relatively simple functions, and $m$ is the number of the simple functions.

Then, we consider the following MOP:

$$\min_{x \in D} f(x) = (f_1(x), \cdots, f_m(x))^T. \tag{3}$$

Let $\hat{x} \in D$ be a Pareto optimal solution of problem (3) if there does not exist an $x \in D$ such that $f_i(x) \leq f_i(\hat{x})$ for all $i = 1, \cdots, m$ and $f_{i_0}(x) < f_{i_0}(\hat{x})$ for some $i_0$. The set of all Pareto optimal solution in $D$ is denoted as $E(f, D) \subset D$. Then, we have the following proposition about the relationship between the optimal solution of problem (1) and the Pareto optimal solutions of problem (3).

**Theorem 1:** Let $\bar{x}$ be an optimal solution of problem (1), then $\bar{x} \in E(f, D)$ of problem (3).

**Proof:** Suppose $\bar{x} \notin E(f, D)$, according to the definition of the Pareto optimal solution, there exists $z \in D$ such that $f_i(z) \leq f_i(\bar{x})$ for all $i = 1, \cdots, m$ and $f_{i_0}(z) < f_{i_0}(\bar{x})$ for some $i_0$. We have that

$$F(z) = f_1(z) + \cdots + f_m(z) < f_1(\bar{x}) + \cdots + f_m(\bar{x}) = F(\bar{x}).$$

That is, $\bar{x}$ is not an optimal solution of problem (1). This implies that the assumption is wrong. Hence, for the optimal solution $\hat{x}$ of problem (1), we have $\hat{x} \in E(f, D)$.

**Theorem 2:** Let $h_i(t)$ be a monotonic increasing function, $(i = 1, \cdots, m)$, we consider the following MOP:

$$\min_{x \in D} h(x) = (h_1(f_1(x)), \cdots, h_m(f_m(x)))^T. \tag{4}$$

Its Pareto optimal solution set $E(h, D)$ satisfies $E(f, D) = E(h, D)$ as shown in [13].

According to Theorem 1 and Theorem 2, we can obtain the extended solution set $E(h, D)$ by solving a relatively simple MOP and then search the final optimal solution $\hat{x}$ of problem (1) from $E(h, D)$.

2.2 Optimization of Each Sub-objective Function Using Tchebycheff Approach

The Tchebycheff approach [13] focuses on the following problem:

$$\min_{x \in D} g(x|w) = \max_{1 \leq i \leq m} \{ w_i f_i(x) \} \tag{5}$$

where $w = (w_1, \cdots, w_m)^T$ is a weight vector. For each Pareto optimal solution $\bar{x}$ of problem (3), there is at least one weight vector $w$ such that $\bar{x}$ is the optimal solution of problem (5). Therefore, one is able to obtain different Pareto optimal solutions by solving the optimization problems defined above by the Tchebycheff approach with the different weight vectors.

However, the function $g(x|w)$ is a non-smooth function. We therefore propose an alternative optimization method to solve it. Specifically, suppose $x^t$ is the optimal solution obtained by implementing the $t$th optimal procedure. Then, the optimization problem of $(t+1)$th optimal procedure can be formulated as follows:

$$\min_{x \in D} g_{t+1}(x|w) = w_1 f_1(x) \tag{6}$$

s.t. $w_1 f_1(x) > (1 - \varepsilon) w_{I_2} f_{I_2}(x^t)$

where $I_1 = \arg \max_{1 \leq i \leq m} w_i f_i(x^t)$, $I_2 = \arg \max_{1 \leq i \leq m, i \neq I_1} w_i f_i(x^t)$ and $\varepsilon$ is a positive constant, $I_1$ and $I_2$ are the indexes of the weighted objective with the fist and second maximum value, respectively.

For $m = 2$, the search procedure can be described as shown in Fig. 1. Suppose the initial solution $x_0$ in the objective space is point $B$ and a given weight vector $(w_1, w_2)$,
we expect to search the optimal point A. From Eq.(6), the current optimization problem is

$$\min_{x \in D} g_1(x, w) = w_2 f_2(x) \quad \text{s.t.} \quad w_2 f_2(x) > (1-\varepsilon) w_1 f_1(x) \quad (7)$$

The search space in objective space of problem (7) can be described as region I in Fig. 1. A new solution $x^1$ is obtained as follows:

$$x^{1+(1)} = x^{1+1},$$

$$x^{1+1} = x^{1+0} = x^{1+0} + \alpha^{1+1} \nabla g(x^{1+1}) \approx \nabla g(x^{1+1})^T x + b.$$

Since $g(x^{1+1}) > (1-\varepsilon) w_2 f_2(x^{1+1})$, the positive scalar $\alpha^{1+1}$ can be computed as $\alpha^{1+1} = \frac{w_1 f_1(x^{1+1}) - (1-\varepsilon) w_2 f_2(x^{1+1})}{\nabla g(x^{1+1})^T \nabla g(x^{1+1}) + C}$, where $C = 0.1$ is a constant which makes the denominator not equal to 0.

However, if $g(x^{1+1})$ is non-differentiable at location $x^{1+1}$, the non-differentiable direct search algorithms, (e.g Hooke-Jeeves Algorithm [15], Powell Algorithm [16]) could be used to simulate the search direction $p^{1+1}$ of downstream operator. In this paper, we utilize the Powell algorithm. The search direction $p^{1+1}$ of downstream operator is represented by the arrow with the solid line in Fig. 2. Therefore, the new location $x^{1+1}$ of $i$th water stream is obtained as follows:

$$x^{1+1} = x^{1+1} + p^{1+1}, \quad i = 1, \ldots, N \quad (9)$$

### 3.2 Penetration Operator

The water penetration is a unity of necessity and contingency. Necessity means most of water streams will penetrate towards the nearby lowest location. Additionally, it is found that the higher the speed of the water stream is, the lower its permeability is. Accordingly, the speed of the water stream penetration is adaptively controlled by its downstream speed. Contingency refers to a little of water runs around all over the place. Thus, a small portion of solutions are operated by a random perturbation to simulate the contingency of water penetration. The heuristic search direction simulating the water penetration is marked as the arrow with the dotted line in Fig. 2.

---

**Figure 1.** Illustration of alternative optimization procedure, where the coordinate of points A, B and C are $(f_1(x_A), f_2(x_A))$, $(f_1(x_0), f_2(x_0))$, $(f_1(x_1), f_2(x_1))$, respectively.

**Figure 2.** The search procedure of SWA for a water stream where $g(x|w)$ is differentiable at location $x^i_1$, the search direction of downstream operator can be formulated as:

$$p^i_1 = -\alpha^i_1 \nabla g(x^i_1|w) = -\alpha^i_1 w_1 \nabla f_1(x^i_1) \quad (8)$$

where $I_1$ is defined as Eq.(6), and $\nabla f_1(x^i_1)$ is the gradient of $f_1(x)$ at location $x^i_1$. The positive scalar $\alpha^i_1$ is called the step length. To compute $\alpha^i_1$, we have linear approximation about $g(x|w)$ at point $x^i_1$, i.e.

$$g(x|w) \approx \nabla g(x^i_1|w)^T x + b.$$
3.2.1 Adaptive Cooperative Heuristic Search

This search is analogous to the phenomenon that the scale of water penetration is within a certain region. Thereby, a neighborhood of rth water stream is defined as the K water streams with the K closest weight vectors to $w$. For the rth water stream, a nearby lowest location which is found by its neighbor water streams is denoted as $x^*_r$, $i = 1, \cdots , N$. Then, the new location of water stream $i$ after perturbation can be formulated as:

$$x^{i+1}_s = x^i_s + \lambda^*_i(x^s_i - x^D_s)$$

(10)

where $\lambda^*_i$ called the speed of penetration is adaptively controlled by the speed of water downstream $p_i$. That is,

$$\lambda^*_i = 0.1r_1 \cdot e^{-\|p_i\|}$$

(11)

where $\|p_i\|$ is the norm of $p_i$ and $r_1$ is a random number generated by a uniform random number generator in $[0, 1]$, denoted as rand. In general, when $x^i_s$ is close to the extreme points, $\|p_i\|$ is relatively small. Equation (11) implies that the smaller of $\|p_i\|$ is, the higher of the speed of penetration is. As shown in Fig. 2, the location $P2$ is close to the local minimum. The penetration dominates the search procedure. It facilitates the solution escapes from the local minimum and avoids the premature convergence in the algorithm. However, when $x^i_s$ is not close to the extreme points, Equation (11) leads to the conclusion that the speed of penetration is relatively small. Thus, the deterministic search direction dominates the search procedure. The positions $P1$ and $P3$ tally with the above situation in Fig. 2. Obviously, it is better for searching minimum optimal solution.

3.2.2 Random Perturbation Heuristic Search

A random heuristic search is performed on a small portion of water streams to simulate the contingency of water penetration. Suppose a random search is performed on $x^D_s$. Every component of $x^D_s$ is perturbed with perturbation probability $p$. If $x^D_{s,l}$, the lth component of $x^D_s$, is selected to perturb, it is reassigned a random number in $[lb(l), ub(l)]$, where $lb, ub$ are the lower bound and upper boundary, respectively. Hence, a new solution is generated as:

$$x^{l+1}_{i,l} = \begin{cases} \text{lb}(l) + r_2(\text{ub}(l) - \text{lb}(l)) & \text{if rand < p} \\ x^D_{i,l} & \text{otherwise} \end{cases}$$

(12)

with $l = 1, \cdots , n$, where $r_2$ is a random number generated from rand. Obviously, the random perturbation facilitates maintaining the diversity of solutions and overcoming premature. As a result, the SWA can be summarized in the next sub-section.

3.3 The Summarized Procedure of SWA

At each fluxion $t$, SWA maintains:

- $N$ points $x^1_i, \cdots , x^N_i \in D$, where $x^i_t$ is the current location of the water stream $i$;
- The lowest locations $x^{l_1}_i, \cdots , x^{l_n}_i \in D$, where $x^{l_i}_i$ is the lowest location to the water stream $i$ which is found by its neighbor water stream;
- The objective values $f(x^{l_i}_i), i = 1, \cdots , N$.

Then the algorithm works as algorithm 1.

4 Experimental Simulation

We conducted SWA on an application example and two benchmark problems to evaluate the performance of SWA in comparison with the existing counterparts.

4.1 Experiment 1

The regularization methods is one of the hot topics in machine learning. The $L_p$ regularizer can be formulated as:

$$\min_x F(x) = \frac{1}{M} \sum_{i=1}^M (Y_i - A^T_i x)^2 + \lambda \|x\|_p$$

(13)

where $M$ is the number of the observations, $A_i = (a_{i1}, \cdots , a_{in})^T$ is drawn from a unknown distribution, $x = (x_1, \cdots , x_n)^T$ is a vector of $n \times 1$, $\|x\|_p$ is the norm of $x$, $\lambda$ is the tuning parameter and $n$ is the number of the variable. As $p \leq 1$, the second term of (13) is a nondifferentiable nonconvex function. It cannot be solved with the differentiable algorithms. We transform the problem into the following MOP:

$$\begin{cases} f_1(x) = \frac{1}{M} \sum_{i=1}^M (Y_i - A^T_i x)^2 \\ f_2(x) = \|x\|_p \end{cases}$$

(14)

and then solve it by using SWA. From theorem 1 we have that the optimal solution of problem (13) will be a Pareto optimal solution of problem (14). It is easy to choose the optimal solution of problem (13) from the Pareto optimal solutions of problem (14). Please note that we need not consider the tuning parameter $\lambda$.

Xu et al. proposed an iteration algorithm to solve $L_{1/2}$ regularizer in [18] and Tibshirani proposed Lasso algorithm to solve $L_1$ regularizer [19]. In this paper, we compare SWA with the iteration algorithm on $L_{1/2}$ regularizer and Lasso algorithm on $L_1$ regularizer to evaluate the performance of SWA. We consider the variable selection application example which is also used in [19] when studying the sparsity of
Algorithm 1: SWA Algorithm

**input:**
- $N$: the number of the water streams;
- $N$ uniform weight vectors: $w^1, \ldots, w^N$;
- $K$: the number of the neighborhood of each water stream;
- $p$: the perturbation probability.

**output:** All the lowest locations $X^L$.

*Initialization:*
- The number of the water streams $N$.
- The maximum number of the fluxion $MF$.
- The perturbation probability $p$.
- The number of the neighborhood of each water stream $K$.
- Uniform weight vectors $w^1, \ldots, w^N$.

For each $i \in \{1, \ldots, N\}$

- Initialization:
  - $X^0_i = \{x^0_1, \ldots, x^0_N\}$ from the decision space $D$.
  - Set $X^L = X^0$.

For $t \leftarrow 1$ to $MF$

- for $i \leftarrow 1$ to $N$

  - Downstream Operator:
    - Apply the downstream operator on $x^t_i$ to produce a location $x^{D}_i$ by Eq. (9);
  - Penetration Operator:
    - Let $r_3$ be a random number from $\text{rand}$.
    - if $r_3 > 0.1$ then
      - The new solution $x^{t+1}_i$ is generated by Eq. (10);
    - else
      - The new solution $x^{t+1}_i$ is generated by Eq. (12);
  - Update the lowest location:
    - foreach $j \in B(i)$ do
      - if $g(x^{t+1}_i | w^j) > g(x^{t}_j | w^j)$ then
        - $x^{L}_j = x^{t+1}_i$ and $f(x^{t+1}_j) = f(x^{t+1}_i)$;
      - end
    - end
  - end
- $t = t + 1$;

The parameter of SWA in experiment 1 are given as follow:

- The number of the water streams $N = 50$
- $N$ weight vectors are uniformly selected from the unit circle in the first octant;
- The number of the neighborhood of each water stream $K = 5$;
- The perturbation probability $p = 0.1$;
- The maximum number of the fluxion $MF = 50$ for 1/2 regularizer and $MF = 30$ for 1 regularizer.

When the absolute value of the element of $x$ is smaller than 0.001, we consider it as zero. The number of the nonzero elements of $x$, denoted as $\text{Deg}(x)$, measures the sparsity of $x$. Fig. 3 plots the final solution obtained by SWA for problem (14) on the first dataset. The left panel is for $L_{1/2}$ and the right panel is for $L_1$ regularizer. There are many solutions obtained in a single run of SWA. Specifically, the left panel of the figure shows that the Pareto front is piecewise concave. The left parts of Pareto solutions with $\text{Deg}(x) = 3$ can make a high reconstruction accuracy. The middle parts with $\text{Deg}(x) = 2$ play a balance rule between accuracy and sparsity. The right parts with $\text{Deg}(x) = 1$ is the sparser solutions for the problem.

**Figure 3.** The final solutions obtained by SWA for problem (14) with different regularizer on the first dataset.

Lasso in this paper. 100 datasets consisting of 100 observations are simulated from the following liner model:

$$Y = A^T \beta + \sigma \varepsilon$$

where $\beta = (3, 1.5, 0, 0, 2, 0, 0, 0)$, $A^T = (a_1, \ldots, a_S)$, $\sigma = 3$ and $\varepsilon$ is random error which is drawn from the standard normal distribution plus 30% outliers from the standard Cauchy distribution. We assume that each $a_i$ obeys normal distribution and the correlation between $a_i$ and $a_j$ is $\rho^{(|i-j|)}$ with $\rho = 0.5$. Since $x$ is an estimate of $\beta$, we assume $-1 \leq x \leq 5$ with $i = 1, \ldots, 8$.

The parameter of SWA in experiment 1 are given as follow:

- The number of the water streams $N = 50$
- $N$ weight vectors are uniformly selected from the unit circle in the first octant;
- The number of the neighborhood of each water stream $K = 5$;
- The perturbation probability $p = 0.1$;
- The maximum number of the fluxion $MF = 50$ for 1/2 regularizer and $MF = 30$ for 1 regularizer.

When the absolute value of the element of $x$ is smaller than 0.001, we consider it as zero. The number of the nonzero elements of $x$, denoted as $\text{Deg}(x)$, measures the sparsity of $x$. Fig. 3 plots the final solution obtained by SWA for problem (14) on the first dataset. The left panel is for $L_{1/2}$ and the right panel is for $L_1$ regularizer. There are many solutions obtained in a single run of SWA. Specifically, the left panel of the figure shows that the Pareto front is piecewise concave. The left parts of Pareto solutions with $\text{Deg}(x) = 3$ can make a high reconstruction accuracy. The middle parts with $\text{Deg}(x) = 2$ play a balance rule between accuracy and sparsity. The right parts with $\text{Deg}(x) = 1$ is the sparser solutions for the problem.
The average number of correctly identified zero elements (CAN in brief) over the 100 tests is recorded. Besides, a set of solutions are obtained by SWA in a single run. The proportion of the solution with different sparsity are also recorded. Table 1 gives the result of Xu’s algorithm and SWA on $L_{1/2}$ regularizer. The proportion of the minority solutions with $Deg(x) > 3$ is not recorded in this table. From Table I, we can see that the CAN of majority solutions obtained by SWA is higher than the result obtain by Xu’s algorithm. This shows that SWA is more efficient and robust than Xu’s algorithm. Table II gives the results of Lasso algorithm and SWA on $L_1$ regularizer. From Table II we can see that the CAN of the solutions with $Deg(x) = 2$ and $3$ is greater than the result obtained by Lasso algorithm. It shows that SWA can provide a more accurate solution. In addition, it can also provide some solutions with higher accuracy of the reconstruction due to a set solutions obtained by SWA in one single run.

**Table 1. Results of Xu’s Algorithm and SWA on $L_{1/2}$ Regularizer**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Deg</th>
<th>Proportion</th>
<th>CAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Xu’s Algorithm</td>
<td>—</td>
<td>—</td>
<td>4.6</td>
</tr>
<tr>
<td>SWA</td>
<td>3</td>
<td>0.34</td>
<td>4.81</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.31</td>
<td>4.92</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>0.25</td>
<td>4.90</td>
</tr>
</tbody>
</table>

**Table 2. Results of Lasso Algorithm and SWA on $L_1$ regularizer**

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Deg</th>
<th>Proportion</th>
<th>CAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lasso</td>
<td>—</td>
<td>—</td>
<td>4.01</td>
</tr>
<tr>
<td>SWA</td>
<td>5</td>
<td>0.21</td>
<td>2.78</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>0.24</td>
<td>3.63</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0.15</td>
<td>4.52</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>0.09</td>
<td>4.79</td>
</tr>
</tbody>
</table>

4.2 Experiment 2

In experiment 2, because of space limitations, we only give the simulation results of SWA on the following single objective benchmarks and compare it with the PSwarm\(^1\) [20] and EA [9].

---

\(^1\)http://www.norg.uminho.pt.
Table 3. Comparative results obtained by the SWA, EA, and PSwarm, respectively, where $n$ is the dimension of the search space, $\text{num}_\text{fun}$ denotes the average number of function evaluations in the 20 runs, $\text{best}$ refers to the best function values in the 20 runs, and $\text{mean}$ is the average value of the best function values in the 20 runs.

<table>
<thead>
<tr>
<th>Instances</th>
<th>$n$</th>
<th>SWA $\text{num}_\text{fun}$</th>
<th>EA $\text{num}_\text{fun}$</th>
<th>PSwarm $\text{num}_\text{fun}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>best</td>
<td>mean</td>
<td>best</td>
</tr>
<tr>
<td>SF1</td>
<td>10</td>
<td>3,000</td>
<td>0</td>
<td>12,000</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>5,000</td>
<td>0</td>
<td>100,000</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>10,000</td>
<td>0</td>
<td>300,000</td>
</tr>
<tr>
<td>SF2</td>
<td>10</td>
<td>3,000</td>
<td>0</td>
<td>12,000</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>5,000</td>
<td>0</td>
<td>100,000</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>10,000</td>
<td>0</td>
<td>300,000</td>
</tr>
</tbody>
</table>

Figure 4. The objective value with the best result versus $\text{num}_\text{fun}$ obtained by SWA, EA and PSwarm for SF1 with $n = 10, 50, 100$.

5 Conclusion

We have proposed an objective decomposition technique to transform a complex optimization problem into an MOP. Subsequently, we have presented an effective SWA algorithm to solve this MOP. Numerical simulations have shown the promising results of the proposed approach in comparison with the existing counterparts.

Acknowledgment

The work described in this paper was supported by the NSFC grant: 61272366, and by Faculty Research Grant of Hong Kong Baptist University with the Project Code: FRG2/08-09/122.

References


