Particle Swarm Optimization with Reduction for Global Optimization Problems

Michiharu Maeda and Shinya Tsuda

Abstract—This paper presents an algorithm of particle swarm optimization with reduction for global optimization problems. Particle swarm optimization is an algorithm which refers to the collective motion such as birds or fishes, and a multi-point search algorithm which finds a best solution using multiple particles. Particle swarm optimization is so flexible that it can adapt to a number of optimization problems. When an objective function has a lot of local minimums complicatedly, the particle may fall into a local minimum. For avoiding the local minimum, a number of particles are initially prepared and their positions are updated by particle swarm optimization. Particles sequentially reduce to reach a predetermined number of them grounded in evaluation value and particle swarm optimization continues until the termination condition is met. In order to show the effectiveness of the proposed algorithm, we examine the minimum by using test functions compared to existing algorithms. Moreover we examine the influence of best value on the initial number of particles for our algorithm is discussed.

Keywords—Particle swarm optimization, Global optimization, Metaheuristics, Reduction.

I. INTRODUCTION

FOR optimization problem, it is difficult to obtain an optimal solution and it requires an immense amount of time. Metaheuristics have been a focus of attention for this situation since they are not dependent on a specific problem [1]. Metaheuristics are optimization approaches which make use of the best solution improved iteratively to the next search. Metaheuristics involve, for example, genetic algorithm (GA), differential evolution (DE), and particle swarm optimization (PSO). GA is a search algorithm which carries out the genetic operation of selection, crossover, and mutation [2]. DE adopts mutation as a weighted sum of a base vector and a difference vector. An individual selected from the population becomes the basic vector and the difference between a pair of individuals becomes the difference vector [3]. PSO is a multi-point search algorithm using multiple candidate solution called particles and performs the solution search by sharing huge amounts of information in each particle. It is a plain and useful algorithm without complicated calculation, and is so versatile that it can adapt to a lot of optimization problems.

The position of the \( i \)-th particle in \( n \)-dimensional space is defined by \( x_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{in})^T \). The velocity of the \( i \)-th particle is represented by \( v_i = (v_{i1}, v_{i2}, v_{i3}, \ldots, v_{in})^T \). Each particle stores the best position previously encountered by particle \( i \) \( p_i = (p_{i1}, p_{i2}, p_{i3}, \ldots, p_{ij}, \ldots, p_{in})^T \), and its evaluation value \( f(p_i) \), the best position of all particles \( g = (g_1, g_2, g_3, \ldots, g_j, \ldots, g_n)^T \), and its evaluation value \( f(g) \). The current position \( x_i^k \) moves to the new position \( x_i^{k+1} \) in addition to the velocity \( v_i^{k+1} \) (the \( i \)-th particle at \( k+1 \)) represented as follows.

\[
v_i^{k+1} = w v_i^k + c_1 r^i (p_i^k - x_i^k) + c_2 r^j (g^k - x_i^k)
\]

where \( w \) is an inertia weight, \( c_1 \) and \( c_2 \) are cognitive and social parameters, respectively, and \( r \) is a random number uniformly distributed in \([0, 1]\).

The new position of particle \( i \) is presented as follows.

\[
x_i^{k+1} = x_i^k + v_i^{k+1}
\]

Figure 1 shows the movement of PSO. \( x_i^k \) is the position of particle, \( v_i^k \) is the velocity of particle, \( g^k \) is the best position of all particles, \( p_i^k \) is the best position previously encountered by particle \( i \), and \( k \) is an iteration. These resultant vectors decide the next velocity of each particle and particles move to the new positions according to Eq. (2). Repeating the similar
procedure, each particle searches for solution space while updating $p$, and $g$.

III. PSO WITH REDUCTION

In this study, we present reduction of PSO. A number of particles is prepared initially and particles are updated in algorithm of PSO. Particles are sequentially reduced to reach a predetermined number based on evaluation value and PSO algorithm of PSO. Particles are sequentially reduced to reach

\begin{equation}
\text{Reduction algorithm of PSO:}
\end{equation}

1. Initialization:
   (1.1) Give parameters $w$, $c_1$, $c_2$, initial number of particles $m_0$, final number of particles $m_f$, maximum iteration $T_{\text{max}}$, and partial iteration $u = T_{\text{max}}/(5(m_0 - m_f + 1))$.
   (1.2) Yield initial velocity $v_i^0$ and initial position $p_i^0$ for each particle at random.
   (1.3) Set $k < 0$, $m = m_0$, $p_i^0 = p_i^0$, and $g_i^0 = g_i^0$, where $s = \arg\min f(p_i^0)$.
2. Update of velocity and position:
   Adapt velocity $v_{i}^{k+1}$ and position $x_{i}^{k+1}$ according to Eqs. (1) and (2).
3. Update $p_i$ and $g_i$:
   If $f(x_{i}^{k+1}) < f(p_i^k)$, then $p_i^{k+1} = x_{i}^{k+1}$, otherwise $p_i^{k+1} = p_i^k$.
   Set $g_{i}^{k+1} = p_i^{k+1}$, where $s = \arg\min f(p_i^{k+1})$.
4. Reduction of particle:
   If $k < u \times q$ and $m > m_f$, then reduce particle $j$ and set $m = m - 1$, where $q$ is a positive integer and $j = \arg\max f(x_{i}^{k+1})$.
5. Termination condition:
   If $k = T_{\text{max}}$, then terminate, otherwise set $k = k + 1$ and go to Step 2.

IV. NUMERICAL EXPERIMENTS

In numerical experiments, we exhibit the proposed algorithm compared to existing algorithms by using six functions in two-dimensional space. Test functions are expressed in the following descriptions.

$F_1$ 2^n minima function:

\begin{equation}
F_1(x) = \sum_{i=1}^{n} [x_i^4 - 16x_i^2 + 5x_i]
\end{equation}

$F_2$ Rastrigin’s function:

\begin{equation}
F_2(x) = 10n + \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i)]
\end{equation}

$F_3$ Levy’s function:

\begin{equation}
F_3(x) = \frac{\pi}{n} \left\{ \sum_{i=1}^{n-1} [(x_i - 1)^2(1 + 10\sin^2(\pi x_{i+1}))] + 10\sin^2(\pi x_1) + (x_n - 1)^2 \right\}
\end{equation}

$F_4$ Schwefel’s function:

\begin{equation}
F_4(x) = -\sum_{i=1}^{n} (x_i \sin(\sqrt{|x_i|}))
\end{equation}

$F_5$ Shubert’s function:

\begin{equation}
F_5(x) = \left\{ \sum_{i=1}^{n-1} i \cos((i + 1)x_1 + i) \right\} + \left\{ \sum_{i=1}^{n-1} i \cos((i + 1)x_2 + i) \right\}
\end{equation}

$F_6$ Shekel’s Foxholes Function:

\begin{equation}
F_6(x) = \left[ \frac{1}{500} + \frac{25}{1 + \sum_{j=1}^{2} \frac{1}{2\sum_{i=1}^{n} (x_i - a_{ij})^6}} \right]^{-1}
\end{equation}

where

$\alpha_{ij} = \left( \begin{array}{cccc}
-32 & -32 & 0 & 0 \\
-32 & -32 & -32 & 16 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
-32 & -32 & -32 & -32 \\
\end{array} \right)$

Tables 1 and 2 show the domain of test functions and the position of minimal solution for each test function, respectively. Figure 2 shows the shape of each function in case of two-dimensional space. Parameters used for experiments are chosen in Table 3 and as follows: $c_1 = 1.4955$, $c_2 = 1.4955$, and $w = 0.729$.

Table 4 shows the minimum in averages of 10000 trials for each of real-coded genetic algorithm (RGA), differential evolution (DE), conventional algorithm (Conv.), and proposed algorithm (Prop.). The number of particle for real-coded...
Fig. 2. Shape of test functions.
genetic algorithm, differential evolution, and conventional algorithm is 20, and the number for proposed algorithm reduces to 50 to 20. It is proven that the proposed algorithm leads to the best position compared to the conventional algorithm. The proposed algorithm is better than existing algorithms.

For reduction, Fig. 3 shows the relationship between the best value and the initial number of particles. We studied the effect that the initial number of particles had on accuracy in reduction. When the initial number is 20, the proposed algorithm becomes the conventional algorithm because there are no particles to be deleted. The best value gradually decreases as the initial number of particles increases. It is found that the proposed algorithm is more effective on the complicated functions given in this experiments.

V. CONCLUSIONS

In this paper, we have presented an algorithm of particle swarm optimization with reduction for improving the solution search accuracy. We examined the minimum value by using test functions to show the effectiveness of the proposed algorithm and the influence of best value on the initial number of particles for our algorithm. As a result, our algorithm had a superiority in comparison with existing algorithms for the complicated functions given in this paper. For the future works, we will study more effective techniques of our algorithm.

TABLE IV
EXPERIMENTAL RESULTS

<table>
<thead>
<tr>
<th>Function</th>
<th>Parameter</th>
<th>RGA</th>
<th>DE</th>
<th>Conv.</th>
<th>Prop.</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>−1.50 × 10⁻³</td>
<td>−1.56 × 10⁻¹</td>
<td>−1.55 × 10⁻¹</td>
<td>−1.57 × 10⁻¹</td>
<td></td>
</tr>
<tr>
<td>F2</td>
<td>3.32 × 10⁻⁵</td>
<td>6.55 × 10⁻³</td>
<td>2.69 × 10⁻⁵</td>
<td>1.99 × 10⁻⁵</td>
<td></td>
</tr>
<tr>
<td>F3</td>
<td>3.55 × 10⁻⁴</td>
<td>3.55 × 10⁻⁴</td>
<td>3.55 × 10⁻⁴</td>
<td>3.55 × 10⁻⁴</td>
<td></td>
</tr>
<tr>
<td>F4</td>
<td>−7.80 × 10⁻⁴</td>
<td>−7.80 × 10⁻⁴</td>
<td>−7.80 × 10⁻⁴</td>
<td>−7.80 × 10⁻⁴</td>
<td></td>
</tr>
<tr>
<td>F5</td>
<td>−1.86 × 10⁻⁴</td>
<td>−1.86 × 10⁻⁴</td>
<td>−1.86 × 10⁻⁴</td>
<td>−1.86 × 10⁻⁴</td>
<td></td>
</tr>
<tr>
<td>F6</td>
<td>5.91</td>
<td>2.10</td>
<td>1.39</td>
<td>1.01</td>
<td></td>
</tr>
</tbody>
</table>

TABLE II
POSITION OF MINIMAL SOLUTION FOR EACH TEST FUNCTIONS

<table>
<thead>
<tr>
<th>Function</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>−5.0 ≤ x₁ ≤ 5.0</td>
</tr>
<tr>
<td>F2</td>
<td>−5.0 ≤ x₁ ≤ 5.0</td>
</tr>
<tr>
<td>F3</td>
<td>0.0 ≤ x₁ ≤ 4.0, 0.0 ≤ x₂ ≤ 6.0</td>
</tr>
<tr>
<td>F4</td>
<td>−500.0 ≤ x₁ ≤ 500.0</td>
</tr>
<tr>
<td>F5</td>
<td>−2.0 ≤ x₁ ≤ 2.0</td>
</tr>
<tr>
<td>F6</td>
<td>−60.0 ≤ x₁ ≤ 60.0</td>
</tr>
</tbody>
</table>

TABLE III
PARAMETERS

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Trials</td>
<td>1000</td>
</tr>
<tr>
<td>Minimum population</td>
<td>20</td>
</tr>
<tr>
<td>Dimensions (n)</td>
<td>*</td>
</tr>
</tbody>
</table>

REFERENCES

(a) $F_1$: 20th minima function

(b) $F_2$: Rastrigin’s function

(c) $F_3$: Levy’s function

(d) $F_4$: Schwefel’s function

(e) $F_5$: Shubert’s function

(f) $F_6$: Shekel’s Foxholes function

Fig. 3. Best value and initial number of particles for each function.