Learning in Particle Swarm Optimization

Shailendra S. Aote, M.M.Raghuwanshi, L.G.Malik

Abstract—This paper presents particle swarm optimization based on learning from winner particle (PSO-WS). Instead of considering gbest and pbest for particle position update, each particle considers its distance from immediate winner to update its position. Only winner particle follow general velocity and position update equation. If this strategy performs well for the particle, then that particle updates its position based on this strategy, otherwise its position is replaced by its immediate winner particle’s position. Dimension dependent swarm size is used for better exploration. Proposed method is compared with CSO and CCPSO2, which are available to solve large scale optimization problems. Statistical results show that proposed method performs well for separable as well as non separable problems.

Index Terms—Large Scale Optimization, Particle Swarm Optimization, Winners Strategy.

I. INTRODUCTION

As many researchers are attracted towards the field of optimization, many optimization methods starting from single objective optimization to multi-objective optimization, unimodal optimization to multimodal optimization, are proposed in past years. Optimization means problem solving in which one seeks to minimize or maximize the objective function in presence of the constraints. Difficulty in solving the problem increases with the increase in dimension size.

Particle swarm optimization (PSO) is global optimization method proposed initially by James Kennedy and Russell Eberhart [1] in 1995. The particle swarm concept originated as a simulation of simplified social system. When a flock of birds travel in the air to find the food, they maintain their velocity and position, so that they never collide each other. There is no central coordinator for the movement of flock. Movement takes place based on cooperation among each bird. Due to its simple equation and involvement of less parameter, it becomes one of the promising algorithms for solving optimization problems.

Each particle is treated as a point in a D dimensional space. The ith particle is represented as

\[ X_i = (X_{i1}, X_{i2}, \ldots, X_{iD}) \]

The best previous position (the position giving the best fitness value) of any particle is recorded and represented as

\[ P_i = (P_{i1}, P_{i2}, \ldots, P_{iD}) \]

The index of the best particle among all the particles in the population is represented by the symbol g. The rate of the position change (velocity) for particle i is represented as

\[ V_i = (V_{i1}, V_{i2}, \ldots, V_{iD}) \]

The velocity and position of the particles are updated according to the following equation,

\[ X_{id} = X_{id} + V_{id} \quad (1) \]

\[ V_{id} = w*V_{id} + c1*rand()*(P_{id}-X_{id}) + c2*rand()*(P_{gd}-X_{id}) \quad (2) \]

rand() is a random function gives value in the range [0,1]. The first part of the equation (1) talks about previous velocity; whereas the second part is the “cognition” part that represents the private thinking of the particle. The third part is the “social” part that represents the collaboration among the particles. Therefore, equation (1) is calculate the particle’s new velocity according to its previous velocity and the distances of its current position from its own best experience (position) and the group’s best experience. Particle moves toward a new position according to equation (2). The performance of each particle is measured according to a predefined fitness function, which is related to the problem to be solved.

These basic PSO equations are updated by researchers. The inertia weight w is brought into the equation (1) as shown in equation (3) by Yuhui Shi in 1998 [8]. This w helps in balancing the global search and local search. It can be a positive constant or even a positive linear or nonlinear function of time.

\[ V_{id} = w*V_{id} + c1*rand()*(P_{id}-X_{id}) + c2*rand()*(P_{gd}-X_{id}) \quad (3) \]

Initially value 0.7 was proposed for w that is further considered as dynamic for better exploration in the beginning and for fine exploitation at the end. Lots of PSO variants are proposed from its formation to till date. Efforts are put towards increase in efficiency and convergence speed. Both the things are rarely achieved in a single algorithm. Low dimensional problems are easier to solve, where as complexity increase as the increase in dimensions and modality. A niching method is introduced in EAs to locate multiple optimal solutions [2]. Distance based LIPS model [3], a memetic PSO [4], Adaptive PSO [5], Fractional PSO [6], AGPSO [7], CSO [33] and many other techniques are proposed to handle higher dimensional and multimodal problems. In spite of these techniques, trapping in local minima and the rate of convergence are two unavoidable problems in PSO and all other EAs. George I. Evers proposed a RegPSO [13], where problem of stagnation is removed by automatically triggering the swarm regrouping. Efforts are taken to solve multimodal problems. To solve higher dimensional problems, variants cooperative co evolution strategies like CPSO-SK and CPSO-HK [14], CCPSO [15], CCPSO2 [16] are proposed.

Following the idea in [33], where losers are updated base on winners. A pair wise is competition is takes place between the particles. Different pairs among the particle are randomly formed. The problem in this strategy is that the loser in one pair may be winner for other pair. So, in this paper, instead of forming the pair randomly, we introduced the concept of immediate winner for each particle. Each particle will update the position based on immediate winners.

Rest of the paper is organized as follows. Section II represents the PSO with Learning. Experimental results and analysis is given into Section III, followed by Conclusion in section IV.

Shailendra S. Aote, Research Scholar, Dept. of CSE, GHRCE Nagpur, India, Mobile N: 9096896565
M.M.Raghuwanshi, Professor, Dept of CT, YCCE, Nagpur
L.G.Malik, Professor, Dept of CSE, GHRCE, Nagpur

ISSN: 2278 – 1323
All Rights Reserved © 2015 IJARCET
3730
II. PSO WITH LEARNING

In order to solve optimization problems, a new PSO with learning strategy is defined here. General PSO considers gbest and pbest to update the position in each iteration. Instead of considering pbest and gbest, we consider immediate winner particle for each. In each iteration, it first evaluates the swarm to calculate the fitness. Sort the swarm in increasing order of the fitness. The best particle follow the general velocity and position update equation, where the third part of velocity update equation is absent because, the best particle itself is a gbest. Other particles consider their immediate winner particle for position update. Immediate winner is termed as given: if i is a current position, then i-1 is considered as winner position. This strategy is more promising as no randomness is involved to define winner as given in [33]. The velocity update equation for particle i is,

\[ \text{vel}(i) = w \times \text{vel}(i) + \Phi \times \text{rand} \times (\text{vel}(i-1) - \text{vel}(i)) \]

Where \( \Phi \) is clerk’s coefficient that controls the swarm and \( w \) is the inertia value, which is ranging from 0.9 to 0.4. If the new position of the particle is better, that particle enters into next generation otherwise its position is replaced by its winner particle’s position.

The proposed algorithm is as follows

**START**

Swarm size = Number of Dimensions
Initialize position and velocity the swarm.
fit = Evaluate the swarm.

**While** termination criteria is not satisfied **do**

Sort the swarm
The best particle follows same velocity and position update equation as per eq. (3) and (2).

for \( j = 2 \) to size of swarm

\[ \text{vel}(j) = w \times \text{vel}(j) + \Phi \times \text{rand} \times (\text{vel}(j-1) - \text{vel}(j)) \]

pos( \( j \) ) = pos ( \( j \) ) + vel ( \( j \) )

end for

fit1= Evaluate the swarm
for \( j = 2 \) to size of swarm

if (fit1( \( j \) ) < fit ( \( j \) ))

Update the position of particle

else

pos(j) =pos (j-1);

end if

end for

**End While**

**End Start**

Computational complexity of PSO-WS is \( O(mn) \), where \( m \) is swarm size and \( n \) is the search dimensions.

III. EXPERIMENTAL RESULTS & ANALYSIS

Proper parameter setting plays very important role in solving the optimization problem. One of the most important parameter is the swarm size. With the small swarm size, convergence is very fast but may cause premature convergence without exploring the search space. It doesn’t mean that swarm size is large, which may cause slower convergence speed. Fig 1 shows the performance of f2 and f6 to decide the swarm size for 1000 dimensions.

50 independent runs are taken for each function and its mean, median, standard deviation and best value is noted for comparison. For each independent run, the maximum number of FEs, as recommended in, is set to 5000 * \( n \), where \( n \) is the search dimension of the test functions. In the comparisons between different statistical results, two-tailed \( t \)-tests are conducted at a significance level of \( \alpha = 0.05 \). The performance is compared with CSO and CCPSO-2, which are known as the state of the art algorithm for large scale optimization problems. Performance is evaluated on 100, 500 and 1000 dimensions on CEC-08 test suit [32] and shown the in table I, II and III respectively.
TABLE I. COMPARISON OF PSO-WS WITH CSO AND CCPSO2 ON 1000 DIMENSIONS

<table>
<thead>
<tr>
<th>1000D</th>
<th>PSO-WS</th>
<th>CSO</th>
<th>CCPSO2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEC08</td>
<td>7.89E-17</td>
<td>6.08E+00</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC09</td>
<td>1.50E+01</td>
<td>1.50E+01</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC10</td>
<td>2.09E-01</td>
<td>1.50E+01</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC11</td>
<td>7.89E-17</td>
<td>6.08E+00</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC12</td>
<td>-1.93E+00</td>
<td>-1.50E+01</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>w/1/NC</td>
<td>6/0/1</td>
<td>3/0/4</td>
<td>4/0/3</td>
</tr>
<tr>
<td>w/1/NC for Std. Deviation</td>
<td>5/1/1</td>
<td>4/0/3</td>
<td>4/0/3</td>
</tr>
</tbody>
</table>

TABLE II. COMPARISON OF PSO-WS WITH CSO AND CCPSO2 ON 5000 DIMENSIONS

<table>
<thead>
<tr>
<th>5000D</th>
<th>PSO-WS</th>
<th>CSO</th>
<th>CCPSO2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEC08</td>
<td>9.28E-17</td>
<td>6.57E-23</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC09</td>
<td>9.10E+00</td>
<td>2.60E+00</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC10</td>
<td>5.18E+01</td>
<td>1.19E+00</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC11</td>
<td>3.92E-01</td>
<td>2.19E+01</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>CEC12</td>
<td>1.93E-01</td>
<td>2.19E+01</td>
<td>7.73E-14</td>
</tr>
<tr>
<td>w/1/NC</td>
<td>6/0/1</td>
<td>3/0/4</td>
<td>4/0/3</td>
</tr>
<tr>
<td>w/1/NC for Std. Deviation</td>
<td>5/1/1</td>
<td>4/0/3</td>
<td>4/0/3</td>
</tr>
</tbody>
</table>

TABLE III. COMPARISON OF PSO-WS WITH CSO AND CCPSO2 ON 10000 DIMENSIONS

<table>
<thead>
<tr>
<th>10000D</th>
<th>PSO-WS</th>
<th>CSO</th>
<th>CCPSO2</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEC08</td>
<td>7.08E-17</td>
<td>6.08E+00</td>
<td>5.18E-13</td>
</tr>
<tr>
<td>CEC09</td>
<td>4.13E+01</td>
<td>4.15E+01</td>
<td>7.82E+00</td>
</tr>
<tr>
<td>CEC10</td>
<td>6.97E+01</td>
<td>1.01E+01</td>
<td>3.38E+00</td>
</tr>
<tr>
<td>CEC11</td>
<td>9.73E+03</td>
<td>1.19E+01</td>
<td>7.82E+00</td>
</tr>
<tr>
<td>CEC12</td>
<td>3.85E+06</td>
<td>2.26E+01</td>
<td>1.19E+01</td>
</tr>
<tr>
<td>w/1/NC</td>
<td>6/0/1</td>
<td>3/0/4</td>
<td>4/0/3</td>
</tr>
<tr>
<td>w/1/NC for Std. Deviation</td>
<td>5/1/1</td>
<td>4/0/3</td>
<td>4/0/3</td>
</tr>
</tbody>
</table>

RESULTS SHOW THAT PSO-WS OUTPERFORMS ON MEAN AS WELL AS STANDARD DEVIATION FOR MOST OF THE FUNCTION, WHERE STATISTICAL RESULTS ARE SIGNIFICANT.

IV. CONCLUSION & FUTURE DIRECTION

In this paper, a new strategy based on immediate winner particle is proposed. After sorting the fitness of particles, each particle updates its position, based on the particle, which has better fitness value. If new position is better as compared to previous position, then it is accepted, otherwise its position is replaced with winner particle’s position. To have a better exploration, dimension dependant swarm size is used. The proposed algorithm is compared with two state of the art for large scale optimization known as CSO and CCPSO2 on CEC-08 test suit. Statistical results show that proposed method performs extremely well on most of the functions.

Future work is to provide theoretical proof of convergence for the proposed method. To test the scalability, proposed Algorithm can be implemented up to 5000 dimensions. This method can be used to solve multiobjective optimization problems.

REFERENCES


ISSN: 2278 – 1323 All Rights Reserved © 2015 IJARCET 3732


