Gbest-guided Group Search Optimizer Algorithm

Seababrat Ghosh  
M.Tech Student, Dept of CSE, Pondicherry University  
seababrataghsh@gmail.com

Kalyan Nandi  
M.Tech Student, Dept of CSE, Pondicherry University  
kalyan.rkmv@gmail.com

Rayees Ahmad Dar  
M.Tech Student, Dept of CSE, Pondicherry University  
darrayes@gmail.com

Abstract

Group Search Optimizer (GSO) is a population-based optimization approach inspired by animal searching behaviour and group living theory. The traditional GSO suffers from the drawback of having slow convergence, poor exploration and exploitation ability. In this paper, an improved version of the GSO is proposed by modifying the solution search equation to incorporate the information of the global best solution (gbest). This improves both the exploration and exploitation. We call this method the gbest-guided GSO (GGSO). Twenty benchmark functions are used to evaluate the performance of the proposed algorithm. Experimental results prove the fact that the GGSO is able to achieve better results than the basic GSO.

Keywords: Group search optimization, Multi-Producer, Gbest-guided GSO, Producer-Srounger model.

1. Introduction

Optimization continues to be one of the principle problems in computer science and hence a lot of research has been directed to find efficient optimization solutions. Inspiration has been drawn from the natural phenomenon to devise new optimization algorithms. Among them are the biological-inspired optimization algorithms, such as genetic algorithm (GA) inspired by the Darwinian law of survival of the fittest [1,2], particle swarm optimization (PSO) inspired by the social behaviour of bird flocking or fish schooling [3,4], ant colony optimization (ACO) inspired by the foraging behaviour of ant colonies [5], Biogeography-Based Optimization (BBO) inspired by the migration behaviour of island species [6], and artificial bee colony (ABC) inspired by the foraging behaviour of honey bee swarm[7]. The animal foraging behaviour has inspired researchers to devise one more such optimization algorithm by adopting the producer – scrounger model [8]. This optimization algorithm is known as Group Search Optimization (GSO) [9] and has been applied successfully to a plethora of optimization problems [10].

GSO imitates the natural phenomenon of producer-scrounger (PS) model for the search of solution by the members of the population. In this regard three types of members are defined as producers, scroungers and rangers. Producers scan for the optimal solutions in the target region. Scroungers follow the producers in finding the solutions. Rangers are meant to escape the local minima by performing random walks. In [11], a group search optimizer strategy for training an ANN was used for diagnosis of breast cancer. Wu et al. [12] presented a multi-objective optimization method, where a group search optimizer with multiple producers (GOMP) was applied to the optimal placement of multi-type Flexible AC Transmission System (FACTS) devices in a power system. He and Li [13] applied an ANN trained with GSO to machine condition monitoring. Recently, many improved GSO algorithms have been proposed such as the Fish Swarm Group Search Optimizer (FSGSO) proposed by Oliveira et al. [14]. It is a hybrid Group Search Optimization technique based on the behaviors of fish swarms. Yan and Shi [15] introduced a novel hybrid algorithm called Group Search Particle Swarm Optimization (GSPSO) based on the Particle Swarm Optimization (PSO) and the Group Search Optimization (GSO), in which the PSO model and the GSO model are used in turns. Junaed et al. [16] proposed a multi-producer GSO (MPGSO) based on multiple producers. Wang et al. [17] proposed a novel multi-objective group search optimizer (NMGSO) for solving the multi-objective optimization problems. Pacifico and Ludermir [18] introduced a novel GSO approach based on cooperative behaviour among groups, called Cooperative Group Search Optimizer (CGSO). Li and Liu [19] proposed a quick group search optimizer (QGSO) for structural optimization. Joshil Raj et al. [20] introduced multi-population group search optimization for function optimization.

Considering the application domain of the GSO [21], it is quite evident that it is a good optimization algorithm. However, there is still an insufficiency in GSO algorithm regarding the solution search equation, which is used to generate new candidate solutions of
GSO algorithm based on the information of previous solutions. The exploration and exploitation are quite necessary for a population based algorithm. However, they are contradictory to each other. Hence in order to achieve satisfactory performance an equilibrium should be found between the two. In this paper, we modify the solution search equation by applying the global best (gbest) solution to guide the search of new candidate solutions in order to improve the exploration and exploitation.

We name the GSO algorithm using the modified solution search equation as Gbest-guided GSO (GGSO) algorithm. Our experiment results tested on numerical function optimization show that GGSO algorithm with appropriate parameter is superior to GSO algorithm in most of the cases. The organization of the paper is as follows: Section 2 discusses the basic GSO, section 3 explains the proposed algorithm namely GGSO. The experimental results of the proposed method in comparison with GA, PSO and basic GSO are given in section 4. Finally in section 5 a conclusion of the study is presented.

2. Group Search Optimizer

In an n-dimensional search space, the i th member at the k th searching iteration has a current position \(X_i^k \in \mathbb{R}^n\), a head angle \(\phi_i^k\) \((\phi_i^1, ..., \phi_i^{n-1}) \in \mathbb{R}^{n-1}\). The search direction of the i th member, which is a unit vector \(D_i^k(\phi_i^k) = (d_{i1}^k, ..., d_{in}^k) \in \mathbb{R}^n\) can be calculated from \(\phi_i^k\) via a polar to Cartesian coordinate transformation:

\[
d_{ij}^k = \prod_{q=1}^{j-1} \cos(\phi_q^k)
\]

\[
d_{ij}^k = \sin(\phi_{ij-1}^k) \prod_{q=j}^{n-1} \cos(\phi_q^k) \quad (j=2, ..., n-1)
\]

\[
d_{in}^k = \sin(\phi_{i(n-1)}^k).
\]

At each member location, the fitness function is evaluated. Among the group members, that member with the best fitness value (minimum value) will be selected as the producer. The producer performs scavenging. Scavenging is performed laterally by randomly selecting one point at zero degree, one point at the right hand side and one point at the left hand side of the producer as given by Eq. (2-4) respectively.

\[
X_x = X_p^k + r_1{l_{max}} D_p^k(\phi^k) \quad (2)
\]

\[
X_r = X_p^k + r_1{l_{max}} D_p^k(\phi^k + r_2\theta_{max}/2) \quad (3)
\]

\[
X_l = X_p^k + r_1{l_{max}} D_p^k(\phi^k - r_2\theta_{max}/2) \quad (4)
\]

Where \(r_1 \in \mathbb{R}^1\) is a random number with standard normal distribution (mean 0 and standard deviation 1) and \(r_2 \in \mathbb{R}^1\) is a random sequence in the range \((0, 1)\).

The producer will then find the best point with the best resource (fitness value). If the best point has a better resource than its current position, then it will fly to this point. Otherwise it will stay in its current position. The producer will also turn its head to a new angle:

\[
\phi_i^{k+1} = \phi_i^k + r_2\alpha_{max}
\]

Where \(\alpha_{max}\) is the maximum turning angle. If the producer cannot find a better area after \(a\) iterations, it will turn its head back to zero degree.

All scavengers will join the resource found by the producer, performing scavenging strategy:

\[
X_i^{k+1} = X_i^k + r_3 \circ (X_p^k - X_i^k) \quad (6)
\]

Where \(r_3 \in \mathbb{R}^n\) is a uniform random sequence in the range \((0, 1)\). Operator “\(\circ\)” is the Hadamard product or the Schur product, which calculates the entry wise product of the two vectors. If a scavenger finds a better location than the current producer and other scavengers, then it will switch as producer in the next iteration.

The remaining members (i.e., rangers) will ranging through random walks \([22]\) using following eq. (7)

\[
X_i^{k+1} = X_i^k + l_i D_i^k(\phi^{k+1}) \quad (7)
\]

Where

\[
l_i = a \cdot r_{1\text{max}} \quad (8)
\]

When a member escapes from the search space bounds, it will turn back to its previous position inside the search space \([23]\).

3. Gbest-guided GSO Algorithm

As mentioned earlier, for a population-based optimization algorithm exploration and exploitation are the two important factors affecting the performance. Exploration is the ability to investigate the various unknown regions in the solution space to discover the global optimum. Exploitation is the capability to find better solutions by using the knowledge of the previous good ones \([24, 25]\). However the two run contradictory to each other and a need to find a balance point is
Begin  
Initialize: Randomly initialize positions $X_i$ and head angles $\phi_i$ of all members.  
Calculate: Calculate the fitness values of initial members  
While termination conditions are not met do  
For each members $i$ in the group do  
	Find new search solution using the following eq.  
	$$X_i^{k+1} = X_i^k + \omega_1 \left( X_i^k - X_d^k \right) + \phi_i \left( \gamma_j - X_i^k \right)$$  
Select Producer: Find the producer of the group.  
	Producer Strategy: (i) Scan at three random points in the scanning field using eq. (2) to (4). (ii) Move to the best point if it has a better resource, otherwise stay at the current position and turn the head to a new angle using eq. (6). (iii) If a better area cannot be found then turn the head back to zero degree using (eq.7).  
Scrounger Strategy: Select a number of group members (normally 80% of the members) as scroungers. Then perform scrouning using (eq.8).  
Ranger Strategy: The rest of the members (rangers) will perform ranging.  
	Calculate the fitness value of each member from this group.  
End For  
End While  
End

necessary for achieving good optimization performance.

In the basic GSO algorithm, the producer tries to search in three directions which is simulated by animals’ foraging food using their eyes. Rotation of the viewing angles and calculation of direction vectors consume a large amount of time. Besides, accumulating evidence and experimental data indicate that the producer’s angle search strategy has a limited role in searching for better points around local optimal region. According to the solution search equation of GSO algorithm described by Eq. (2-4 & 6-7), the new candidate/member solution is generated by moving the old solution (scrounger) towards another solution (producer) selected from the population. The new candidate solution is not promising to be a solution better than the previous one. On the other hand, in Eq. (6), the coefficient $r_3 \in \mathbb{R}^n$ is a uniform random sequence in the range (0, 1). The solution search Eq. (2-4 & 6-7) is random enough for exploration but poor at exploitation. Inspired by PSO [4], which, in order to improve the exploitation, takes advantage of the information of the global best (gbest) solution to guide the search of candidate solutions, we introduce a new solution search equation as follows

$$X_i^{k+1} = X_i^k + \omega_1 \left( X_i^k - X_d^k \right) + \phi_i \left( \gamma_j - X_i^k \right)$$

where the third term in the right-hand side of Eq. (9) is a new added term called gbest term, the coefficient $\omega_1$ is a uniform random number in [-1,1] and $X_d^k$ is a random individual in the population, $\gamma_j$ is the jth element of the global best solution, $\phi_i$ is a uniform random number in [0, C], where C is a nonnegative constant. According to Eq. (9), the gbest term can drive the new candidate solution towards the global best solution, therefore, the modified solution search equation described by Eq. (9) can increase the exploitation of GSO algorithm. Although the solution search equation of GGSO algorithm described by Eq. (9) is similar to PSO [26, 27], GGSO algorithm still preserves the main characteristics of GSO algorithm, which can distinguish GGSO algorithm from PSO. GGSO algorithm is clearly different from PSO, because GGSO algorithm does a comparison between the new candidate solution and the old solution and then saves the better one, while PSO does not involve such selection procedure. The pseudo code for the GGSO algorithm is presented in Algorithm 1.
Eighteen Benchmark Function, Where N is the Dimension of the Function, $f_{min}$ is the Minimum Value of the Function And $\mathbb{R}^n$

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Functions, $f$</th>
<th>$N$</th>
<th>$S$</th>
<th>$f_{min}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sphere</td>
<td>$f_1(x) = \sum_{i=1}^{n} x_i^2$</td>
<td>30</td>
<td>$[-100,100]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Schwefel2.22</td>
<td>$f_2(x) = \sum_{i=1}^{n}</td>
<td>x_i</td>
<td>+ \prod_{i=1}^{n}</td>
<td>x_i</td>
</tr>
<tr>
<td>Schwefel 1.2</td>
<td>$f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$</td>
<td>30</td>
<td>$[-100,100]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Schwefel2.21</td>
<td>$f_4(x) = \max_i(</td>
<td>x_i</td>
<td>, 1 \leq i \leq n)$</td>
<td>30</td>
</tr>
<tr>
<td>Rosenbrock</td>
<td>$f_5(x) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$</td>
<td>30</td>
<td>$[-30,30]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Step</td>
<td>$f_6(x) = \sum_{i=1}^{n} (</td>
<td>x_i + 0.5</td>
<td>)^2$</td>
<td>30</td>
</tr>
<tr>
<td>Rastrigin</td>
<td>$f_7(x) = \sum_{i=1}^{n} (x_i^2 - 10 \cos(2\pi x_i) + 10)^2$</td>
<td>30</td>
<td>$[-5.12,5.12]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Ackley</td>
<td>$f_8(x) = -20 \exp \left(-0.2 \sqrt{\sum_{i=1}^{n} x_i^2} - \exp(\sum_{i=1}^{n} \cos(2\pi x_i)\right) + 20 + e$</td>
<td>30</td>
<td>$[-32,32]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Penalized</td>
<td>$f_9(x) = 0.1 \sin^2(\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 (1 + \sin^2(2\pi x_i) + 1 + \sin^2(2\pi x_i) + \sum_{i=1}^{n} u(x_i, 5, 10, 4)$</td>
<td>30</td>
<td>$[-50,50]^n$</td>
<td>0</td>
</tr>
<tr>
<td>Shekel Foxholes</td>
<td>$f_{10}(x) = \left[\frac{1}{500} + \sum_{i=1}^{25} \frac{1}{\sum_{i=1}^{n} (x_i - a_{ij})^6}\right]^{-1}$</td>
<td>2</td>
<td>$[-65.536,65.536]^n$</td>
<td>1</td>
</tr>
<tr>
<td>Hump</td>
<td>$f_{11}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^2$</td>
<td>2</td>
<td>$[-5,5]^n$</td>
<td>$-1.0316285$</td>
</tr>
<tr>
<td>Branin</td>
<td>$f_{12}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10 \left(1 - \frac{1}{8\pi}\right) \cos x_1 + 10$</td>
<td>2</td>
<td>$[-5,10] \times [0,15]$</td>
<td>$0.398$</td>
</tr>
<tr>
<td>Goldstein-Price</td>
<td>$f_{13}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)]$</td>
<td>2</td>
<td>$[-2,2]^n$</td>
<td>3</td>
</tr>
<tr>
<td>Hartman 3</td>
<td>$f_{14}(x) = -\sum_{i=1}^{4} c_i \exp - \sum_{i=1}^{3} a_{ij} (x_i - p_{ij})^2$</td>
<td>3</td>
<td>$[0,1]^n$</td>
<td>$-3.86278$</td>
</tr>
<tr>
<td>Hartman 6</td>
<td>$f_{15}(x) = -\sum_{i=1}^{4} c_i \exp - \sum_{i=1}^{6} a_{ij} (x_i - p_{ij})^2$</td>
<td>6</td>
<td>$[0,1]^n$</td>
<td>$-3.32237$</td>
</tr>
<tr>
<td>Shekel 5</td>
<td>$f_{16}(x) = \sum_{i=1}^{5} [(x - a_i)(x - a_i)^r + c_i]^{-1}$</td>
<td>4</td>
<td>$[0,10]^n$</td>
<td>$-10.1532$</td>
</tr>
<tr>
<td>Shekel 7</td>
<td>$f_{17}(x) = \sum_{i=1}^{7} [(x - a_i)(x - a_i)^r + c_i]^{-1}$</td>
<td>4</td>
<td>$[0,10]^n$</td>
<td>$-10.4029$</td>
</tr>
</tbody>
</table>
Shekel 10
\[ f_{10}(x) = \sum_{i=1}^{10} [(x - a_i)(x - a_i)^T + c_i]^{-1} \]

4. Experimental Studies

4.1. Benchmark Problems and Experimental Setup
In this section, the Gbest-guided GSO algorithm is applied to minimize a set of 9 scalable benchmark functions of dimensions \( N = 30 \) and a set of 9 functions of lower dimensions \( N = 2, 3, 4, 6 \). Three different kinds of functions are selected: unimodal functions \( (f_1 \text{to} f_6) \), multimodal functions \( (f_7 \text{to} f_9) \), low-dimensional multimodal functions \( (f_{10} \text{to} f_{18}) \). These functions are listed in Table I. The algorithms are implemented in MATLAB environment. The experiments have been done on a PC (Intel(R) Core(TM) i3-2310M CPU @2.10GHz, 3GB RAM) with Windows 7 OS.

### TABLE III
Comparison of GGSO with GA, PSO, Basic GSO on Eighteen Benchmark Function

<table>
<thead>
<tr>
<th>Function</th>
<th>GA</th>
<th>PSO</th>
<th>GSO</th>
<th>GGSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>3.1711</td>
<td>3.6927x10^{-37}</td>
<td>1.9481x10^{-8}</td>
<td>2.105336x10^{-9}</td>
</tr>
<tr>
<td>F2</td>
<td>0.5771</td>
<td>2.9168x10^{-24}</td>
<td>3.7039x10^{-5}</td>
<td>7.880331x10^{-8}</td>
</tr>
<tr>
<td>F3</td>
<td>9749.9145</td>
<td>1.1979x10^{-3}</td>
<td>5.7829</td>
<td>3.07034x10^{-8}</td>
</tr>
<tr>
<td>F4</td>
<td>7.9610</td>
<td>0.4123</td>
<td>0.1078</td>
<td>8.476871x10^{-2}</td>
</tr>
<tr>
<td>F5</td>
<td>338.5616</td>
<td>37.3582</td>
<td>49.8359</td>
<td>15.55377</td>
</tr>
<tr>
<td>F6</td>
<td>3.6970</td>
<td>0.1460</td>
<td>1.6000x10^{-2}</td>
<td>5.012362x10^{-10}</td>
</tr>
<tr>
<td>F7</td>
<td>0.6509</td>
<td>20.7863</td>
<td>1.0179</td>
<td>1.332268x10^{-13}</td>
</tr>
<tr>
<td>F8</td>
<td>0.8678</td>
<td>1.3404x10^{-3}</td>
<td>2.6548x10^{-3}</td>
<td>1.377703x10^{-6}</td>
</tr>
<tr>
<td>F9</td>
<td>0.1681</td>
<td>5.0519x10^{-2}</td>
<td>4.6948x10^{-5}</td>
<td>2.873383x10^{-14}</td>
</tr>
<tr>
<td>F10</td>
<td>0.9989</td>
<td>1.0239</td>
<td>0.9980</td>
<td>0.9980</td>
</tr>
<tr>
<td>F11</td>
<td>-1.0298</td>
<td>-1.0160</td>
<td>-1.031628</td>
<td>-1.031628</td>
</tr>
<tr>
<td>F12</td>
<td>0.4040</td>
<td>0.44040</td>
<td>0.3979</td>
<td>0.3978874</td>
</tr>
<tr>
<td>F13</td>
<td>7.5027</td>
<td>3.0050</td>
<td>3.0</td>
<td>3.00</td>
</tr>
<tr>
<td>F16</td>
<td>-5.1653</td>
<td>-7.5439</td>
<td>-6.09</td>
<td>-10.1532</td>
</tr>
<tr>
<td>F17</td>
<td>-5.4432</td>
<td>-8.9439</td>
<td>-7.4022</td>
<td>-10.4029</td>
</tr>
<tr>
<td>F18</td>
<td>-4.9108</td>
<td>-8.9439</td>
<td>-7.4022</td>
<td>-10.5320</td>
</tr>
</tbody>
</table>

The initial head angle \( \phi_0 \) of each individual is set to be \( (\pi/4, \ldots, \pi/4) \). The maximum pursuit angle \( \theta_{max} \) is \( \pi/a^2 \). The maximum turning angle \( \alpha_{max} \) is set to be \( \theta_{max}/2 \). The constant ‘a’ is given by \( \text{round} (\sqrt{n} + 1) \) where \( n \) is the dimension of the search space. The maximum pursuit distance \( l_{max} \) is calculated from the eq.5. The population size is 48 and 80% members were selected as scroungers. For GA algorithm Crossover Rate is 0.9 and Mutation Rate 0.1 and Population Size is 50. The PSO algorithm we executed is the standard one. The parameters were given by default setting of the toolbox: the acceleration factors \( c_1 \) and \( c_2 \) were both 2.0 and a decaying inertia weight \( \omega \) starting at 0.9 and ending at 0.4 was used. The population of 50 was used in the PSO algorithm.
4.2. Experimental Result and Analysis

In this section, we compare the proposed Gbest-GSO with the GA, PSO and basic GSO algorithms. Table II shows the results for all test functions of the Gbest-GSO and other three algorithms taken as the mean value of all the functions. The best values according to the empirical analysis are bolded. As mentioned in Section III, the new searching equation of GGSO algorithm plays an important role in controlling the exploration and exploitation of the new candidate solution search. The empirical analysis showed that the proposed MPGSO method achieved a better performance than standard GSO for twelve of the eighteen ($f_3$ to $f_9$, $f_{12}$ and $f_{15}$ to $f_{19}$) functions. Three functions ($f_{10}$, $f_{11}$, $f_{13}$) gave the same result. It also outperformed PSO for sixteen functions ($f_3$ to $f_{18}$). GGSO always gives better result than GA for all the function. PSO gives better performance than GSO for sphere function and schwefel2.22 function. Fig. 1 to Fig. 3 show the variation on the optimal or best fitness value for basic GSO and Gbest-guided GSO method according to number of function evaluations.

![Graph showing comparison between GSO and GGSO](image)

**Fig. 1 Optimized value of function $f_1$ to $f_4$ and $f_6$ to $f_9$**

![Graph showing comparison between GSO and GGSO](image)

**Fig. 2 Optimized value of function $f_{11}$ and $f_{14}$ to $f_{18}$**

![Graph showing comparison between GSO and GGSO](image)

**Fig. 3 Optimized value of function $f_5$, $f_{10}$ $f_{12}$ and $f_{13}$**
5. Conclusions

In this paper, group search optimizer (GSO) algorithm was studied. Observing that the solution search equation of GSO algorithm is good at exploration but poor at exploitation, we proposed an improved GSO algorithm called Gbest-guided GSO (GGSO) algorithm, which takes advantage of the information of global best solution to guide the search of new candidate solutions in order to improve the exploitation. The GGSO is evaluated on eighteen well known benchmark functions for 2 to 30-dimension search space problems and compared to standard GSO, PSO and GA algorithms. Experimental results show that the GGSO can achieve better results than standard GSO and the other tested algorithms for most of the cases.

6. References


