HONEY BEE FORAGING PSO ALGORITHM WITH SELF-ADJUSTING NEIGHBORHOOD SIZE

MUHAMMAD RASHID  
National University of Computer and Emerging Sciences, A.K. Brohi Road, Sector H-11/4, Islamabad, Pakistan

ABDUL RAUF BAIG  
National University of Computer and Emerging Sciences, A.K. Brohi Road, Sector H-11/4, Islamabad, Pakistan

ABSTRACT  
In this paper we present an extension to the Honey Bee Foraging PSO (HBF-PSO) algorithm. This algorithm is modeled after the food foraging behavior of the honey bees and performs a collective foraging for fitness in promising neighborhoods in combination with individual scouting searches in other areas. The strength of the algorithm lies in its continuous monitoring of the whole scouting and foraging process with dynamic relocation of the bees if more promising regions are found. The algorithm has the potential to be useful for optimization problems of multi-modal nature. The extension we propose allows HBF-PSO to automatically adjust the neighborhood size during execution which results in improved performance. The details of the algorithm are presented followed by experimental results on some commonly used multi-modal benchmark test functions. We present a comparison of both versions of HBF-PSO with NichePSO and SPSO.

INTRODUCTION  
Swarm based optimization algorithms have witnessed considerable attention lately. Optimization algorithms based on the behavior of birds (PSO) and ants (ACO) are well known and widely used [8]. PSO is known to be suitable for function optimization problems where the goal is to search for the optimum value of a function. Multimodal function optimization is a variant of the general optimization problem in which the problem has more than one optimum or near optima. Other researchers have attempted to address this class of problems using swarm based approach and have shown that variants of canonical PSO algorithm are able to solve these problems. Some of the main efforts of swarm based solutions for multi-modal problems include [5-7, 9, 12-14].

In this paper we present an extension to the HBF-PSO algorithm [3], and experimentally show that it can be applied to multimodal problems with greater success as compared to the original HBF-PSO. The main characteristic of the HBF-PSO is the incorporation of three new features in the PSO algorithm. We propose simultaneous exploration and exploitation of the fitness landscape along with a blackboard type of information sharing mechanism. These features allow parallel searching for fitness peaks and relocation of search towards promising areas. These features are derived from a model of the food foraging behavior of honey bees. It has been observed that scout bees gather information and the bees at the hive get asynchronous updates by means of waggle dances of the scout bees.
bees. If a promising flower patch is discovered, bees are sent from the hive for foraging. The quantity of allocated bees is proportional to the quality of patch.

In HBF-PSO, several swarms of particle-bees forage (search for peak fitness) in a collective and coordinated manner. A swarm assembles in a promising region and then moves towards the top of the peak. Furthermore, there are scout particle-bees that are placed randomly and perform a local search. The foraging and scouting are allowed to continue for a limited duration, sufficient for a swarm to forage its region of allocation (converge on peak fitness). Corresponding to the waggle dance of the bees at the hive, a blackboard is maintained which may be updated asynchronously whenever some useful information is available either from the scouting or foraging particle-bees. Particle-bees are reallocated after analyzing the information on the blackboard.

Some researchers have tried in the past to mimic the behavior of honey bees in their algorithms. Abbass [1, 2] has presented the Marriage in honey Bees Optimization (MBO) algorithm which is modeled after the mating flight of queen bees. Pham [15, 16] has proposed the Bees algorithm which is inspired by the food foraging behavior of the honey bees. The main characteristic of Bees algorithm is the static sampling of fitness landscape. Examples of other applications based on honey bee behavior include a routing algorithm [18]. Several entomologists and sociologist have also tried to model the behavior of honey bees, for example,[11].

A few researchers have proposed PSO based solutions for multimodal problems. In [9] it is proposed that the particles be clustered into sub-swarms using a k-means clustering algorithm. The sub-swarms then use the centers of the clusters as substitute for personal bests or neighborhood best. In [13] and [14] a method is introduced in which after finding a good candidate solution it is isolated and other particles are kept away from this area by stretching the fitness landscape. A small population is generated around the isolated particle to allow finer search whereas the remaining population searches the rest of the landscape. In [6] a nbest PSO algorithm is proposed in which the average of the positions of the particles closest to a particle is taken as its neighborhood best. The neighborhood is defined as the closest particles of the population in terms of Euclidean distance. In [5] a parallel niching algorithm called NichePSO is presented which locates multiple optimal solutions through the use of sub-swarms and a guaranteed convergence sub-swarm optimization algorithm GCPSO. In [7] the scalability of NichePSO is compared with two genetic algorithm niching techniques, sequential niching and deterministic crowding. The authors concluded that NichePSO has an upper hand on more complex problems. In [12] the concept of species is used for determining neighborhood best values. The algorithm is called Species PSO (SPSO). It divides the swarm into subpopulations based on their similarities. Within each species a seed is identified which serves as the neighborhood best for that species.

The rest of the paper is organized as follows. In the next section we presents the food foraging process of the honey bees followed by a description of the HBF-PSO algorithm. We then present our extension to the HBF-PSO algorithm which is followed by the experimentation conducted and the result obtained along with a discussion of results. In the end we conclude our current work and give future directions of research.
FOOD FORAGING BEHAVIOR OF HONEY BEES

In this section, we present our model of the food foraging behavior of honey bees. For the development of the HBF-PSO algorithms the actual foraging process is simplified as follows. At the start of the food foraging process scout bees are sent out from the hive to look for flower patches. These bees return with information about the quantity, quality and direction of flower patches and communicate this information to the other bees by means of a waggle dance. Additional bees are then recruited based on available information to forage the discovered flower patches, with more bees being attracted towards richer flower patches. The scouting and foraging process continues simultaneously during the whole harvesting season.

Our model of the above biological process is as follows. The colony is supposed to consist of a fixed number of bees. Initially all the bees go out in random directions to find flower patches. After a certain time they all return to the bee hive and analyze the collected information about the richness (quality and quantity) of the patches that they have discovered. Several bees go out to forage the discovered flower patches and other bees go out to scout the remaining region randomly. After spending a certain time in foraging/scouting, all of the bees again return to the bee hive and analyze the collected information. These alternate bouts of analysis and foraging/scouting are repeated again and again until the whole area has been sufficiently searched. These two phases are explained in more detail below.

Book-Keeping & Analysis Phase: Some aspects of the history of scouting and foraging are recorded and analyzed at the hive (corresponding to the waggle dance). During the analysis phase, the richness of each flower patch is compared with the richness of other patches. The bees decide which of the patches need to be foraged thoroughly in the next round. Each selected patch is assigned a separate swarm composed of a few bees. The remaining bees are assigned to perform random search.

Richness record of each patch being foraged is kept and monitored during subsequent analysis. Based on this record a patch may be declared as completely foraged and bees are no longer assigned to that patch. Furthermore, the richer regions are foraged on a priority basis. A current patch may be abandoned on the discovery of a new richer patch. After the foraging of the richer patches is complete the bees may return to this abandoned patch again.

Scouting & Foraging Phase: Each selected flower patch is foraged by a swarm of several bees in a cooperative and collective manner. On the other hand, each scout bee (starting from an arbitrary point) explores its surrounding region locally and without any help from other bees. This foraging and scouting process continues for some prespecified time duration. At the end of that time period the bees gather again at their hive, pool the information and start the analysis phase.

HONEY BEE FORAGING PSO (HBF-PSO) ALGORITHM

Particle swarm optimization (PSO) was originally designed and developed by Eberhart and Kennedy [10] and later extended to include inertia weight in [17]. In PSO the population of candidate solutions is represented by a swarm of particles. Each particle is a point in the N-Dimensional search space. The $i^{th}$
particle in the swarm is represented by its current position \( x_i \), and its current velocity \( v_i \) (both \( x_i \) and \( v_i \) are \( N \)-dimensional vectors). PSO tries to find the optimal solution to the problem by moving the particles and evaluating the fitness of the new position. A particle's position is updated by the following equation.

\[
x_{i}^{t+1} = x_{i}^{t} + v_{i}^{t}
\]

Along with updating the particle's position, its velocity is updated with every iteration as well. The amount by which the velocity is changed is dependent upon three factors; the particle's personal best position, the swarm's best position and the particle's previous velocity. The velocity is updated according to the following equation.

\[
v_{i}^{t+1} = w \times v_{i}^{t} + c_1 \times r_1 \times (p_{i}^{t} - x_{i}^{t}) + c_2 \times r_2 \times (p_{g}^{t} - x_{i}^{t})
\]

Where \( w \) is the inertia weight and \( c_1 \) and \( c_2 \) are constants and \( r_1 \) and \( r_2 \) are random numbers in the range \( [0 \ldots 1] \). \( p_{i} \) is the personal best position of the particle and \( p_{g} \) is the global best position of the swarm. The second part of the above equation is usually referred to as the cognition part because due to this part the particle's new velocity is influenced by its personal best position; whereas the third part is referred to as the social part because due to this part the particle's new velocity is influenced by the swarm's best position.

In HBF-PSO one bee is one particle, i.e. one complete solution of the problem. A fixed number of bees (\( h \)) are created and placed randomly in the fitness landscape. The fitness at each of these points is calculated. The points are sorted on the basis of their fitness and bee swarms are created around each of the \( m \) best points. If any two or more of the best points are near one another then they are considered as belonging to the same region and only one swarm is created for them. In other words, overlapping of swarms in not allowed during their creation. This condition encourages that there may be only one exploitation per promising region.

The creation of swarms means that \( n \) bees have been assigned to each of these swarms. The remaining bees (\( f = h - m - n \)) are used as scouts and placed randomly. These two types of bees are called foraging bees and scouting bees, respectively.

The foraging and scouting bees search for a small, fixed number of iterations in the following manner. The \( n \) foraging bees search for a peak as a swarm in a coordinated and collective manner according to classical PSO algorithm. Meanwhile each of the \( f \) scout bees searches around its lieu of placement alone. This is done according to PSO algorithm without social component.

After a few iterations of simultaneous foraging and scouting a global analysis takes place. The best fitness reported from each of the swarm and the best fitness of each of the scout bees are all sorted and the swarms for the next iteration are determined on the basis of the information. An old swarm is allowed to survive only if its best fitness comes within the first \( m \) best points. If one of these best points has been newly discovered by a scout bee, then a new swarm is created around it. As mentioned before, no overlapping is allowed during the creation of swarms. This eliminates one of the two swarms which have come close to one another (the swarm with the lower fitness is eliminated). This manner of creating swarms gives the algorithm a preference for exploring
the higher fitness regions on a priority basis which usually leads to the discovery of higher peaks first in multi peak environments.

If a swarm has converged on a peak its best fitness value becomes stagnant. This phenomenon is kept under observation for each swarm. If a swarm is not reporting any improvement in fitness during a few consecutive analysis phases, then it is assumed to have reached a peak. The swarm is disbanded and the region around its best fitness is considered as completely foraged. A completely foraged region is then considered taboo and is not included in any further search. The scouting/foraging followed by analysis continues until a stopping condition is fulfilled. This condition can be a prescribed number of cycles or the discovery of a pre-specified number of peaks or sufficient foraging of the search space (for example, if we are unable to find unexplored regions with a fitness greater than 10% of the best fitness) or a combination of these three conditions.

Summary of HBF-PSO Algorithm: The HBF-PSO algorithm can be summarized in the following steps.

1. Initialize all the parameters and generate an initial population of bees.

2. In the initial iteration, all bees are scouting bees. Each bee randomly evaluates the fitness at one place in the search space. In subsequent iterations, there would be several swarms and several scouting bees.

3. At the beginning of each iteration form m swarms of n bees each. This is done in the following manner
   (a) Compile a list of fitness by including the best fitness reported by each of the scout bees and the best fitness from each of the swarms in the previous iteration. Sort this list in descending order.
   (b) Remove the current top fitness from the list. If the retrieved fitness is due to a swarm continue with the old swarm. If it is due to a scout bee, create a new swarm around it. This is done by placing one bee on the position where that fitness has been obtained and placing other \(n-1\) bees randomly in the neighboring region around it. After the first swarm is created, we have to avoid multiple swarms in the same region, for this we check to see if the area of the swarm formation (neighborhood) overlaps any other neighborhood of a swarm created during the current execution of this step. Discard the retrieved fitness in that case and retrieve the next fitness from the list. Also check if the neighborhood overlaps any area declared as completely foraged (step 6 below). Discard the retrieved fitness in this case also and retrieve the next fitness from the list.
   (c) Go to step (b) until m swarms have been created.

4. Allocate remaining bees for random scouting.

5. The bees of each swarm search the region in a collective and cooperative manner while each of the scout bees searches near its lieu of placement on an individual basis. This search continues for a limited number of iterations. The best fitness values from each swarm and the fitness of each scout bee are recorded.

6. Check to see if any swarm has converged and become stagnant. In that case mark the corresponding region as completely foraged because there is no need of further exploration in that area.

7. Go to step 3 until a termination condition is met.
PROPOSED EXTENSION

While creating new swarms in HBF-PSO, a neighborhood size is employed which restricts the initialization of the particles of that swarm to within a specified region. The neighborhood size needs to be accurately defined as it greatly affects the performance of the algorithm. Ideally the neighborhood size should be half of the distance between two closest optima. This is to ensure that the region in which a single swarm is initialized contains only one optimum. We now propose an extension to the HBF-PSO algorithm which allows it to automatically adjust and adapt the neighborhood size during its execution. In this extension once a swarm converges to a solution, we perform a check to determine if the number of peaks found have changed. If so, we recalculate the distance between the closest two peaks and set the neighborhood size equal to half of this distance.

EXPERIMENTS

We performed two sets of experimentations, the first one was performed with the original HBF-PSO algorithm. In the second experiment we measured the performance of the modified HBF-PSO algorithm which incorporates self-adjusting neighborhood size. We also compared the performance of both HBF-PSO variants with other niche based algorithms. For this purpose we used five test functions suggested by Beasley [4] and used in both [5] and [12]. Table 1 gives a list of these functions.

<table>
<thead>
<tr>
<th>Function</th>
<th>Range</th>
<th>Optima</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F1(x) = \sin^5(5\pi x)$</td>
<td>[0,1]</td>
<td>5</td>
</tr>
<tr>
<td>$F2(x) = \exp\left(-2 \log(2) \cdot \left(\frac{x - 0.1}{0.8}\right)^2\right) \cdot \sin^6(5\pi x)$</td>
<td>[0,1]</td>
<td>1</td>
</tr>
<tr>
<td>$F3(x) = \sin^5(5\pi(x^{3/4} - 0.05))$</td>
<td>[0,1]</td>
<td>5</td>
</tr>
<tr>
<td>$F4(x) = \exp\left(-2 \log(2) \cdot \left(\frac{x - 0.08}{0.854}\right)^2\right) \cdot \sin^6(5\pi(x^{3/4} - 0.05))$</td>
<td>[0,1]</td>
<td>1</td>
</tr>
<tr>
<td>$F5(x, y) = 200 - (x^2 + y - 11)^2 - (x + y^2 - 7)^2$</td>
<td>[-6,6]</td>
<td>4</td>
</tr>
</tbody>
</table>

To measure the performance of HBF-PSO, the accuracy and success rate were measured. Accuracy is calculated by taking the average of the fitness difference between all known global optima to their closest peak detected. Success rate is measured by the percentage of runs (out of 50) locating all global optima within the 50 HBF-PSO iteration steps, for the minimum of an expected accuracy 0.0001. To make a fair comparison with SPSO and NichePSO the setup is made as close as possible to both SPSO and NichePSO. HBF-PSO is executed 50 times for each of the test functions. For each run 50 HBF-PSO iterations were allowed with each iteration having 40 PSO iterations (a total of 2000 PSO iterations). The total number of bees used is 30, the number of swarms is set to 5, the number of bees in each swarm is set to 5, and the number of scout bees is 5. For the original HBF-PSO the neighborhood for swarm creation is set to $\pm 0.05$ for F1, F2, F3, F4 and $\pm 0.20$ for F5 in each dimension as specified in [12]. The modified HBF-PSO also used these values to initialize the
neighborhood size but was allowed to modify them with the execution of the algorithm.

RESULTS

Table 2 shows the results of the experiment for the original HBF-PSO as well as those reported for NichePSO [5] and SPSO [12]. All three algorithms were able to achieve an accuracy of 100%. However HBF-PSO produced better accuracy for all of the test function as compared to NichePSO and produced a better accuracy for four test function when compared to SPSO.

Table 2. Results on accuracy after 50 HBF-PSO iterations (2000 PSO iterations) (averaged over 50 runs).

<table>
<thead>
<tr>
<th>Function</th>
<th>Accuracy (HBF-PSO) (mean and std err)</th>
<th>Accuracy (SPSO) (mean and std err)</th>
<th>Accuracy (NichePSO) (mean and std err)</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1.14E-08 ± 2.23E-08</td>
<td>0.00 ± 0.00</td>
<td>7.68E-05 ± 2.20E-04</td>
</tr>
<tr>
<td>F2</td>
<td>1.33E-17 ± 2.61E-17</td>
<td>4.00E-17 ± 2.26E-17</td>
<td>9.12E-02 ± 6.43E-02</td>
</tr>
<tr>
<td>F3</td>
<td>3.33E-16 ± 6.26E-16</td>
<td>3.20E-14 ± 3.20E-14</td>
<td>5.95E-06 ± 4.86E-05</td>
</tr>
<tr>
<td>F4</td>
<td>3.29E-10 ± 6.16E-10</td>
<td>1.72E-07 ± 0.00</td>
<td>8.07E-02 ± 6.68E-02</td>
</tr>
<tr>
<td>F5</td>
<td>2.43E-27 ± 4.76E-27</td>
<td>2.19E-09 ± 2.19E-09</td>
<td>4.78E-06 ± 1.03E-05</td>
</tr>
</tbody>
</table>

Table 3 presents the results obtained for the modified HBF-PSO with self adjusting neighborhood size. It can be seen that the modified HBF-PSO algorithm gave better performance for all five text functions as compared to the original HBF-PSO algorithm. The average neighborhood size for each function is also given. As can be seen, the neighborhood was adjusted according to the problem which resulted in better accuracy.

Table 3. Results on accuracy of modified HBF-PSO after 50 HBF-PSO iterations (2000 PSO iterations) (averaged over 50 runs).

<table>
<thead>
<tr>
<th>Function</th>
<th>Accuracy (HBF-PSO) (mean and std err)</th>
<th>Average neighborhood size</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>3.73E-12 ± 7.14E-12</td>
<td>1.24E-01 ± 1.07E-01</td>
</tr>
<tr>
<td>F2</td>
<td>0.00 ± 0.00</td>
<td>9.49E-02 ± 2.17E-02</td>
</tr>
<tr>
<td>F3</td>
<td>1.66E-15 ± 3.12E-15</td>
<td>1.10E-01 ± 1.10E-01</td>
</tr>
<tr>
<td>F4</td>
<td>1.45E-11 ± 8.53E-18</td>
<td>9.90E-02 ± 5.00E-02</td>
</tr>
<tr>
<td>F5</td>
<td>4.06E-31 ± 2.32E-32</td>
<td>2.04 ± 3.05E-01</td>
</tr>
</tbody>
</table>

CONCLUSIONS

We present an extension to the HBF-PSO algorithm. HBF-PSO is based on simultaneous search of multiple promising regions along with supplemental scouting for better regions. A centralized analysis takes place after a fixed number of iterations and the search priorities are recalculated. The extension we propose enables HBF-PSO to automatically adjust and adapt the neighborhood size at runtime. The neighborhood size is an important parameter which greatly affects the performance of HBF-PSO. By enabling HBF-PSO to automatically adjust it to the search space we were able to achieve better performance. We
also present a comparison with two other well known algorithms, NichePSO and SPSO. We are able to show experimentally that our algorithm exhibits better performance then these two on some famous benchmark function. We are looking to further expand our experimentation to accommodate more test functions and to include comparisons with other algorithms to better elaborate the performance of HBF-PSO.

For future work we note that other researchers are experimenting with PSO in multi-objective environments and a few are even venturing into multi-objective dynamic environments. It would be interesting to see if the advantages of HBF-PSO over classical PSO holds in multi-objective dynamic environments. Thus one direction of our efforts for the future will be to try to cater for multi-objective dynamic environments.

REFERENCES