Ensemble bat algorithm based on Hyper heuristic approach for solving unconstrained optimization problems

Wakas S. Khalaf

1Department of Industrial Management, College of Economics and Administration, University of Baghdad

Article History: Received: 10 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 28 April 2021

Abstract Maintaining convergence and diversification in solving optimization is one of the most important challenges facing metaheuristic algorithms in general and the bat algorithm in particular. Many researchers have suggested some improvements to preserve the ability of the algorithm to find good solutions in a timely manner and also to move away as much as possible from landing on the local optimization zone. In this paper, a hyper-heuristic method was proposed to incorporate the behavior of three optimized algorithms from the bat algorithm. The method is based on the distribution of a specific implementation probability for each used algorithm and then updating this probability iteratively according to the results of each algorithm, and then we use random selection to determine the algorithm used in the current iteration. Some nonlinear models proposed in CEC2005 used to compare the efficiency of the proposed algorithm and compare its results with some state-of-the-art algorithms.

Keywords: Bat algorithm, Ensemble strategy, unconstrained optimization problems, Benchmark function

1. Introduction

Many scientific and engineering problems can be turned into numerical optimization problems for solving functions. A nonlinear unconstrained optimization problem can be defined as follows without sacrificing generality:

\[ \min f(x) \]
\[ s.t. \quad l_i \leq x_i \leq u_i \quad i = 1, 2, ..., n \quad (1) \]

Among them, \( f(x) \) is the objective function, \( x = (x_1, x_2, ..., x_n) \) Í \( \mathbb{R}^n \) is the n-dimensional decision variable, and \( l_i \) and \( u_i \) are the upper and lower bounds of the variable \( x_i \) respectively.

The conventional optimization approach based on gradient knowledge is difficult to find an effective solution for problem (1) because it is typically highly nonlinear. Intelligent optimization algorithms like the genetic algorithm, particle swarm optimization algorithm, differential evolution algorithm, ant colony algorithm, artificial immune algorithm, and others, when compared to conventional optimization methods, are more efficient. are a form of global search method that uses population iteration and has no problem requirements. It is high and does not require the problem's gradient knowledge, and it can converge to the problem's global optimal solution with a high probability. As a result, it's common in unconstrained optimization [1-5].

Yang [9] proposed the Bat Algorithm (BA), a multitude insight enhancement calculation that is utilized to display the characteristic pursuit of bats utilizing ultrasound. The streamlining issue's answer is alluded to as a bat in the pursuit space in BA. A wellness esteem is doled out to each bat. The bats can change the recurrence of their heartbeats, To adjust the best bat at present in the arrangement region as far as volume and outflow rate. Search for something. In taking care of unconstrained advancement issues, BA has been demonstrated to be better than Genetic algorithm (GA) and particle swarm optimization (PSO) calculations [9]. Therefore, BA can be utilized in an assortment of settings, including mathematical improvement, designing enhancement, obliged streamlining, and creation arranging issues.

Nonetheless, like other multitude knowledge streamlining calculations, essential BA likewise has inadequacies, for example, being inclined to fall into neighborhood optima and defer late combination. Focusing on the insufficiencies of fundamental BA, an incorporated cross breed BA is proposed to tackle
the issue of unconstrained advancement. To consolidate different bat calculation renditions and super-heuristic stage disorder, establish the framework for improving the calculation variety, in this way instating the position and speed of bats. To organize the worldwide and nearby inquiry capacities of the calculation, a latency weight is presented in the speed update equation. Play out a Powell search on the flow best bat to accelerate union. The reenactment after effects of a few standard test capacities demonstrate the adequacy of the calculation.

In this paper, we will combine algorithms BA-DTFS proposed in [4] and BA-DTFS proposed in [5] and MBA proposed in [33] to propose a new hybrid algorithm from the bat search algorithms. The merging mechanism will depend on a hyper-heuristic approach to distribute the probability of implementation on each algorithm used, with the development of criteria for comparing the algorithm's results at each stage with some randomly generated values.

The paper is organized as follows. Section 2 describes the basic Bat algorithm with a detail on algorithm and presents the proposed Ensemble of BAT algorithm variants (EBATV) with diserption. Section 3 provides details of benchmarking problems, parameter setting of the algorithms and compares their results. Section 4 provides on results and discussion results.

2. Methodology

2.1 BAT Algorithm

Yang [9] proposed BA, a heuristic intelligent search algorithm inspired by bats’ use of echolocation for predator detection. To solve the problem, first map a single bat to specific points in the search space, then use the position of a single bat in the search space as the fitness function's value. Optimize the objective purpose of the problem, The search process of the algorithm is modeled after that of bats looking for food and flying. The basic steps of the BA algorithm are as follows:

Step 1: Make \( t = 1 \) and initialize the algorithm's parameters.

Step 2: Randomly generate solutions \( x^t_i \) and in the velocity for each solution \( v^t_i \).

Step 3: Determine the fitness value of each solution and the population's best solution.

Step 4: Check to see if the algorithm satisfies the termination condition (whether it reaches the maximum number of iterations). The algorithm ends when it is satisfied, and the best solution is output. If not, proceed to step 5.

Step 5: Update the solution's velocity and position using equations (2), (3), and (4):

\[
\begin{align*}
    f_i &= f_{\min} + (f_{\max} - f_{\min}) \cdot \beta \\
    v^{t+1}_i &= v^t_i + (x^t_i - x^*). f_i \\
    x^{t+1}_i &= v^{t+1}_i + x^t_i
\end{align*}
\]

Where \( f_i \) is the \( i \)-th bat's pulse frequency, \( f_{\min} \) and \( f_{\max} \) are the pulse frequency's minimum and maximum values, respectively, and is a uniformly distributed random number on \([0, 1]\), \( v^{t+1}_i \) and \( v^t_i \) respectively. Is the \( i \)-th bat's flight speed in generations \( t + 1 \) and \( t \), and \( x^t_i \) is the \( i \)-th bat's location in \( t \), \( x^* \). The ideal position of the bat in the current community is \( t \), and the \( i \)-th bat at the position of \( t + 1 \) generation is \( x^{t+1}_i \).

Step 6: Rand1 is a random number generator. If rand1 > \( r_i \) (\( r_i \) is the \( i \)-th bat's pulse frequency), then perturb the current optimal bat location to get a new one, and then replace the old one.

Step 7: Generate a random number rand2. If \( \text{rand1} > A_i \) (\( A_i \) is the pulse intensity of the \( i \)-th bat) and \( f(x_i) < f(x^*) \), then move to the updated position.

Step 8: When the condition of Step 7 is satisfied, the pulse frequency \( r \) and pulse intensity \( A \) are updated according to equations (5) and (6):
Ensemble bat algorithm based on Hyper heuristic approach for solving unconstrained optimization problems

\[ r_i^{t+1} = r_i^0 [1 - e^{(-\gamma \times t)}] \quad (5) \]
\[ A_i^{t+1} = \alpha \cdot A_i^t \quad (6) \]

In the formula, \( r_i^{t+1} \) denotes the \( i \)th bat's pulse frequency at \( t+1 \) generation, \( r_i^0 \) denotes the \( i \)th bat's maximum pulse frequency, \( \gamma > 0 \) denotes the pulse frequency increase factor, \( A_i^{t+1} \) and \( A_i^t \) denote the sound strength of the \( i \)th bat transmitting pulses at generations \( t+1 \) and \( t \), respectively, \( \alpha \in [0, 1] \) the pulse intensity attenuation; otherwise, let \( t = t + 1 \) and return to step 3. The flow of the standard Bat algorithm is as follows.

**Algorithm 1: Standard BAT Algorithm**

Begin
Set the relevant parameters for each bat.
Calculate each solution's fitness values;
While (The stopping condition of the algorithms is satisfied)
Using Eq.4, update the position of each bat.
Assess the new position's fitness requirements;
If the current position's fitness is lower than the previous position's,
Remove the old solution and replace it with the new one;
End If
Choose the best particle and save it;
End While
Provide the best solution;
End

2.2 Ensemble of BAT algorithm variants (EBATV)
In this study, in order to collect multiple BAT variants, a multi-population framework (MPF) was proposed. MPF does not implement the combination of Population into subpopulation (PAP) algorithm through plan for time allocation particles immigration administrators and their strategies [20], but divides the entire PAP, including several indicator subpopulations and reward subpopulations. The indicator subgroups are the same size as the incentive subgroups, but they are much smaller. There is an indicator subgroup for each variant that makes up BAT. Any certain number of generations after the creation of the integration algorithm, the reward subgroup will be adaptively assigned to the BAT best performance variable. In this way, different BAT variants will evolve together, and the variant with the best performance during the evolution will get the most (total) resources.

2.2.1 Constituent BAT variants
In order for EBATV to work better, the constituent variables of BAT must be both strong and have different functions, so that they can assist each other in the evolution process, not just in competing resources. Many studies have shown that using different operators in algorithms is important [80-82]. Three highly efficient BAT variants, namely BA-DTFS [4], BAGW [5], and MBA [33], are used as synthesis algorithms in this study. Attempting to incorporate each variant of BAT into the integration process is impossible. MBA is a generic BAT variant, according to observational analysis, so these three algorithms were chosen as components. It typically dominates other BAT variables when solving the single peak optimization problem, and BAGW is solving some simple problems. Multimodal optimization is a very successful problem. BA-DTFS has shown exceptional success in the optimization of highly complex compounds [2, 24]. The following is a quick rundown of the three BAT variants.

1. BA-DTFS
Triangle-flipping technique in the Bat algorithm (BA-DTFS), initially developed by Cai and et.al [4], is a simple but efficient BAT variant. A new position update strategy is used in BA-DTFS. The position category is listed below.

\[ x_i^{t+1} = x^* + (x_{\text{worst}} - x_i^t) \cdot f_i \quad (7) \]
\[ x_i^{t+1} = x_i^t + (x_{\text{worst}} - x^*) \cdot f_i \quad (8) \]
\[ x_i^{t+1} = x_{\text{worst}} + (x_i^t - x^*) \cdot f_i \quad (9) \]
In the above three triangle inversion strategy, all bats appear in those bats, and their positions have not been determined in the previous generation. (3) And (4). All bats can use triangle rotation strategy to update their speed and position.

2. BAGW
Bat algorithm with Gaussian walk (BAGW) Proposed by Wang and et. al [5] In this variant, Gaussian traces are used instead of the original uniform traces in local turbulence to improve local search capabilities. In addition, in order to maintain a higher pumping pressure, The speed update equation was also changed as a result of the increased strain. Finally, in order to maximize population diversity, the frequency is determined by each size and varies depending on the various bats in our modification. In the regular version, the location and velocity are described below.

\[ v_{i}^{t+1} = (x_{i}^{t} - x^{*}) \cdot f_{i} \]  \hfill (10)

\[ x_{i}^{t+1} = x^{*} + \eta \mu \]  \hfill (11)

The random numbers sampled by a standard Gaussian distribution are used among them. According to the study above, a large number of numbers are distributed in order to maximize the chances of escaping the local optimal value.

3. MBA
Bat calculation has been changed (MBA) The point of Ramli and et a proposition . [33] is to improve the investigation and utilization of the bat calculation to accomplish a quicker assembly speed. By consolidating new versatile size adjustments and new inertial weight changes, this can be refined. Inertial weight influences the speed condition, which thusly influences the whole BA measure. The inertial weight esteem relies upon the speed. Speed can be estimated by the distance between the current best position and the current situation at emphasis t. The inactivity esteem ceaselessly diminishes alongside cycles and merges to emphasis t, which shows that the bat is nearer to acquiring prey (arrangement). Determined as follows:

\[ w_{t} = (t_{max} - t) \cdot \sqrt{(f(x^{t}) - f(x^{*}))^{2}} \]  \hfill (12)

\[ v_{i}^{t+1} = v_{i}^{t} \cdot w_{t} + (x_{i}^{t} - x^{*}) \cdot f_{i} \]  \hfill (13)

2.2.2 Multi-Swarm based ensemble framework
The MPF divides the population into several indicator subgroups (each individual belongs to the constitutive BAT variant) and rewards them. We divide the entire population into three subgroups of indicators and a reward population because the MPF contains three BAT algorithm variants, namely BA-DTFS, BAGW, and MBA. Each generation activates the partition operator. \(P_{1}, P_{2},\) and \(P_{3}\) represent the three indicator subgroups, while \(P_{4}\) represents the incentive subgroup. The index subgroups are all the same size. The indicator subgroup is much smaller than the reward subgroup in terms of size. Let pop represent the entire population. We've got

\[ P = \bigcup_{i=1}^{4} P_{i} \]  \hfill (14)

Let \(NP\) be the size of the \(P\) and \(NP_{i}\) be the size of the \(P_{i}\). \(w_{i}\) indicates the proportion between \(P_{i}\) and \(P\). So we have

\[ NP_{i} = w_{i} \cdot NP \]  \hfill (15)

\[ \sum_{i=1}^{4} w_{i} = 1 \]  \hfill (16)

We just have \(w_{1} = w_{2} = w_{3}\), here. Each indicator subpopulation is assigned to a constituent BAT variant at random, and the reward subpopulation is also assigned to a BAT variant at random. The population partition procedure is done once for each generation. After each number of generations, the algorithm continues, Based on the relationship between the cumulative fitness improvements and the evaluations of the functions consumed, we evaluate the most effective BAT variant (\(f_{best}\)) over the last century.

**Algorithm 2:** Algorithm Framework of EBATV.
1. Initial parameters of EBATV including $ng, NP, wi, MaxG$ and $MaxFes$;
2. Initial the parameters for BA-DTFS, BAGW and MBA;
3. Set $\Delta f_i = 0$ and $\Delta fes_i$ for $i = 1, 2, 3$;
4. Initialize the $P$ randomly distributed in the solution space;
5. Set $NP_i = w_i \ast NP$;
6. Randomly divide $P$ into $P_1, P_2, P_3$ and $P_4$ with respect to their sizes;
7. Randomly select a subpopulation $P_i (i = 1, 2, 3)$ and combine $P_i$ with $P_4$. Let $P_i = P_i \cup P_4$
and $NP_i = NP_i + NP_4$;
8. Set $g = 0$;
9. while $g \leq MaxG$ do
10. $g = g + 1$;
11. Execute BA-DTFS on $P_1$, update $P_1$ and calculate $\Delta f_1$;
12. Execute BAGW on $P_2$, update $P_2$ and calculate $\Delta f_2$;
13. Execute MBA on $P_3$, update $P_3$ and calculate $\Delta f_3$;
14. Combine updated $P_1, P_2$ and $P_3$ into $P$, i.e., $P = \bigcup_{i=1}^{3} P_i$;
15. if $\text{mod}(g, ng) == 0$ then
16. $k = \text{arg}(\max_{i=1,2,3}(\frac{\Delta f_i}{\Delta fes_i}))$;
17. end if
18. Randomly partition $P$ into $P_1, P_2, P_3$ and $P_4$;
19. Let $P_k = P_k \cup P_4$, $k \in \{1, 2, 3\}$;
20. End while

$$j_{best} = \text{arg max}_{i=1,2,3} \left( \frac{\Delta f_i}{\Delta fes_i} \right)$$ (17)

where $\Delta f_i$ is the cumulative function fitness improvements attributed by the $i$th constituent BAT variant over the last $ng$ generations, and $\Delta fes_i$ is the consumed number of function evaluations. The reward subpopulation will be awarded to the highest performing constituent BAT variant for the next $ng$ generations. The above-mentioned best-performing BAT variant determination and reward subpopulation assignment operators are run on a regular basis, with $ng$ being the time. We ensure that the best upgrade velocity approach uses the most computational resources with this definition. Algorithm 2 describes the EBATV algorithm architecture.

3. Experimental study

1. Benchmark problem and comparative algorithms

"Any elevated performance over one class of problems is exactly compensated for in performance over another class," according to the No Free Lunch theorem. In other words, a meta-heuristic can perform

admirably on one set of problems while performing poorly on another set of problems. Without a doubt, Every year, the NFL keeps this area of study active, resulting in improvements to existing methods and the introduction of new meta-heuristics. We used a broad collection of standard benchmark functions, which are listed in Table I, to completely evaluate the output of the EBATV algorithm without coming to a biased conclusion about any specific problems. The ten benchmark functions are divided into three categories: unimodal functions ($F_1$ to $F_7$) and multimodal functions ($F_9$ to $F_{13}$). Despite the fact that this set of benchmark functions has been widely adopted by other researchers [3,] their dimensions are relatively small (up to 30) as compared to those of real-world optimization problems. Figure 1 shows the space of variables and objective function for each problem.

Comparative algorithms

We put EBATV to the test with 10 and 30 variables on the CEC2005 [39] benchmark problems. BAT [9], BA-DTFS [4], BAGW [5], and MBA [33] are some of the more recent algorithms that have been chosen as comparatives. PSO [38], PSO [38], PSO [38], PSO [38] Every comparative algorithm's parameters are identical to those in the original paper. For all algorithms, the population size is 20 and the number of function evaluations is 5000. The outcomes of 30 simulation runs on ten and thirty-dimensional problems.
\[
\begin{array}{cccc}
\text{Test Function} & n & S & f_{\text{min}} \\
F_1(x) = \sum_{i=1}^{n} x_i^2 & 0 & -100,100] & 0 \\
F_2(x) = \sum_{i=1}^{n} |x_i| + \prod_{i=1}^{n} |x_i| & 0 & -10,10] & 0 \\
F_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2 & 0 & -10,10] & 0 \\
F_4(x) = \max_{i} \{ |x_i|, 1 \leq i \leq n \} & 0 & -100,100] & 0 \\
F_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i)^2 + (x_i - 1)^2] & 0 & -100,100] & 0 \\
F_6(x) = \sum_{i=1}^{n} [(x_i + 0.5)^2] & 0 & -30,30] & 12,569. \\
F_7(x) = \sum_{i=1}^{n} ix_i^4 + \text{random}(0,1) & 0 & -100,100] & 0 \\
F_8(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{|x_i|}) & 0 & -100,100] & 0 \\
F_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10] & 0 & -1.28,1.28] & 0 \\
F_{10}(x) = 20 \exp\left(-0.2 \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right) - \exp\left(\cos(2\pi x_i)\right) + 20 + e & 0 & -500,500] & 0 \\
F_{11}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{i}\right) + 1 & 0 & -32,32] & 0 \\
F_{12}(x) = \frac{\pi}{n} \left[10 \sin(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2 [1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2 \right] + \sum_{i=1}^{n} u(x_i, 10, 100, 4) & 0 & -600,600] & 0 \\
y_i = \frac{1 + x_i + 1}{4} u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\
0 & -a < x_i < a \\
k(-x_i - a)^m & x_i < -a \end{cases} & 0 & [ -50, 50] & 0 \\
F_{13}(x) = 0.1 \left[\sin^2(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 [1 + \sin^2(3\pi x_i + 1)] + (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] \right] + \sum_{i=1}^{n} u(x_i, 5, 100, 4) & 0 & -50,50] & 0 \\
\end{array}
\]
Ensemble bat algorithm based on Hyper heuristic approach for solving unconstrained optimization problems

Figure (1): Continued
## Results and discussion results

<table>
<thead>
<tr>
<th>$F$</th>
<th>EBATV</th>
<th>BAT</th>
<th>BA-DTFS</th>
<th>BAGW</th>
<th>MBA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>$F1$</td>
<td>4.256907</td>
<td>5.27E-09</td>
<td>0.010641</td>
<td>3.608328</td>
<td>0.001183</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.85317)</td>
<td>(1.20E-08)</td>
<td>(0.002843)</td>
<td>(1.16555)</td>
<td>(0.00089)</td>
<td></td>
</tr>
<tr>
<td>$F2$</td>
<td>38.91408</td>
<td>1.80E+01</td>
<td>0.024057</td>
<td>7.952319</td>
<td>38.73776</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(13.6195)</td>
<td>(1.14E+01)</td>
<td>(0.005229)</td>
<td>(1.29307)</td>
<td>(39.2677)</td>
<td></td>
</tr>
<tr>
<td>$F3$</td>
<td>5510.18</td>
<td>11456.74</td>
<td>38801.77</td>
<td>14.82813</td>
<td>4556.914</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(8468.54)</td>
<td>(6234.571)</td>
<td>(5446.878)</td>
<td>(1.29307)</td>
<td>(2101.31)</td>
<td></td>
</tr>
<tr>
<td>$F4$</td>
<td>53.16875</td>
<td>9.92344</td>
<td>0.011301</td>
<td>0.001183</td>
<td>17.46107</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(5.37020)</td>
<td>(2.532465)</td>
<td>(0.005229)</td>
<td>(0.00089)</td>
<td>(3.04736)</td>
<td></td>
</tr>
<tr>
<td>$F5$</td>
<td>1.62E+11</td>
<td>27336.74</td>
<td>25.68098</td>
<td>125.1801</td>
<td>71.69537</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(3391.25)</td>
<td>(43265.75)</td>
<td>(36.4322)</td>
<td>(97.1070)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$F6$</td>
<td>5.894768</td>
<td>3.31E-09</td>
<td>0.011301</td>
<td>17.03625</td>
<td>0.00146</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(1.21292)</td>
<td>(5.35E-09)</td>
<td>(0.005229)</td>
<td>(3.61670)</td>
<td>(0.00094)</td>
<td></td>
</tr>
<tr>
<td>$F7$</td>
<td>0.193691</td>
<td>0.566405</td>
<td>0.05396</td>
<td>10.39285</td>
<td>0.234667</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.05675)</td>
<td>(1.698824)</td>
<td>(0.005229)</td>
<td>(2.74419)</td>
<td>(0.09222)</td>
<td></td>
</tr>
<tr>
<td>$F8$</td>
<td>-8360.08</td>
<td>-9476.92</td>
<td>-7143.01</td>
<td>-116.961</td>
<td>-5472.81</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(97.3293)</td>
<td>(533.1439)</td>
<td>(204.1972)</td>
<td>(7.52E+0)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$F9$</td>
<td>204.4521</td>
<td>97.9248</td>
<td>134.7475</td>
<td>56.85804</td>
<td>82.48186</td>
<td></td>
</tr>
<tr>
<td>$F10$</td>
<td>2.800032</td>
<td>2.06E-05</td>
<td>0.0362327</td>
<td>2.656609</td>
<td>3.65361</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.27665)</td>
<td>(2.86E-05)</td>
<td>(0.005515)</td>
<td>(0.20038)</td>
<td>(1.57675)</td>
<td></td>
</tr>
<tr>
<td>$F11$</td>
<td>1.040986</td>
<td>0.0105738</td>
<td>0.102645</td>
<td>0.163409</td>
<td>0.071146</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.01771)</td>
<td>(0.011987)</td>
<td>(0.0122785)</td>
<td>(0.05764)</td>
<td>(0.02811)</td>
<td></td>
</tr>
<tr>
<td>$F12$</td>
<td>31232.32</td>
<td>0.1299941</td>
<td>0.034312</td>
<td>1.447892</td>
<td>12.2892</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(27016.2)</td>
<td>(0.231838)</td>
<td>(0.01729)</td>
<td>(0.68375)</td>
<td>(6.11068)</td>
<td></td>
</tr>
<tr>
<td>$F13$</td>
<td>2.778063</td>
<td>165577.7</td>
<td>0.106449</td>
<td>27.81043</td>
<td>17.1326</td>
<td></td>
</tr>
<tr>
<td></td>
<td>(0.59454)</td>
<td>(127488.9)</td>
<td>(0.026226)</td>
<td>(0.04914)</td>
<td>(17.1326)</td>
<td></td>
</tr>
</tbody>
</table>
Ensemble bat algorithm based on Hyper heuristic approach for solving unconstrained optimization problems

Table (3)

<table>
<thead>
<tr>
<th>F</th>
<th>EBATV</th>
<th>BAT</th>
<th>BA-DTFS</th>
<th>BAGW</th>
<th>MBA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1</td>
<td>0.99795</td>
<td>6</td>
<td>0.99999</td>
<td>0.99826</td>
<td>0.99999</td>
</tr>
<tr>
<td>F2</td>
<td>1</td>
<td>0.20894</td>
<td>9</td>
<td>0.99951</td>
<td>8.39E-01</td>
<td>0.212524</td>
</tr>
<tr>
<td>F3</td>
<td>1</td>
<td>0</td>
<td>2</td>
<td>0.9973</td>
<td>1</td>
<td>0.9173</td>
</tr>
<tr>
<td>F4</td>
<td>1</td>
<td>0.15778</td>
<td>1</td>
<td>0.74071</td>
<td>0.98675</td>
<td>0.723408</td>
</tr>
<tr>
<td>F5</td>
<td>1</td>
<td>0.78042</td>
<td>8</td>
<td>0.99906</td>
<td>0.9542</td>
<td>0.997377</td>
</tr>
<tr>
<td>F6</td>
<td>1</td>
<td>0.99706</td>
<td>4</td>
<td>0.99999</td>
<td>0.99152</td>
<td>0.99999</td>
</tr>
<tr>
<td>F7</td>
<td>1</td>
<td>0.98136</td>
<td>4</td>
<td>0.94550</td>
<td>0.99940</td>
<td>0.997421</td>
</tr>
<tr>
<td>F8</td>
<td>0.670388</td>
<td>0.80213</td>
<td>1</td>
<td>7.61E-01</td>
<td>0.57140</td>
<td>0.435575</td>
</tr>
<tr>
<td>F9</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0.99999</td>
<td>0.99741</td>
<td>0.596571</td>
</tr>
<tr>
<td>F10</td>
<td>1</td>
<td>0.80004</td>
<td>3</td>
<td>0.99999</td>
<td>0.99741</td>
<td>0.81028</td>
</tr>
<tr>
<td>F11</td>
<td>1</td>
<td>0.96347</td>
<td>6</td>
<td>0.99639</td>
<td>0.99426</td>
<td>0.997504</td>
</tr>
<tr>
<td>F12</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>0.99999</td>
<td>0.99999</td>
<td>0.999607</td>
</tr>
<tr>
<td>F13</td>
<td>0.999983</td>
<td>0</td>
<td>1</td>
<td>0.99999</td>
<td>0.99999</td>
<td>1.00E+0</td>
</tr>
</tbody>
</table>

Result of comparative between the algorithms

Table (4)

<table>
<thead>
<tr>
<th>F</th>
<th>EBATV</th>
<th>BAT</th>
<th>BA-DTFS</th>
<th>BAGW</th>
<th>MBA</th>
<th>PSO</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1</td>
<td>0.99979</td>
<td>6</td>
<td>0.99999</td>
<td>0.9972</td>
<td>1</td>
</tr>
<tr>
<td>F2</td>
<td>1</td>
<td>0.65316</td>
<td>3</td>
<td>0.70968</td>
<td>0.9986</td>
<td>7</td>
</tr>
<tr>
<td>F3</td>
<td>1</td>
<td>0.36229</td>
<td>6</td>
<td>0.53052</td>
<td>0.58983</td>
<td>0.99928</td>
</tr>
<tr>
<td>F4</td>
<td>1</td>
<td>0.77029</td>
<td>7</td>
<td>0.89167</td>
<td>0.93064</td>
<td>2</td>
</tr>
<tr>
<td>F5</td>
<td>1</td>
<td>0.92161</td>
<td>8</td>
<td>0</td>
<td>0.99999</td>
<td>1</td>
</tr>
<tr>
<td>F6</td>
<td>1</td>
<td>0.99971</td>
<td>2</td>
<td>0</td>
<td>0.99999</td>
<td>3</td>
</tr>
</tbody>
</table>
Discussion Result

Table (2) shows that the EBATV generated better result than all algorithms in function $F_1$– $F_4$ and $F_6$ the EBATV algorithm get the optimal solution in standard deviations equal to zero that meaning the EBATV algorithm get the optimal solution in all running. in function $F_1$– $F_2$ the best result after EBATV was BA-DTFS algorithm and the worst result in BAT algorithm. And in function $F_3$ the best result after EBATV found by using MBA and the worst result found by using BAT. And in function $F_4$ the best result after EBATV found by using MBA and the worst result found by using BAT, in function $F_5$ the best result found by EBATV algorithm and then BAGW algorithm and worst result found by BA-DTFS, in function $F_6$ the best result after EBATV found by using BA-DTFS and the worst result found by using MBA, in function $F_7$ the best result found by EBATV algorithm and then BAGW algorithm and worst result found by MBA.

Additionally in correlation with the outcomes delivered by EBATV. From Table (2), it is obvious to see that for F9 to F12 of the test benchmark capacities, EBATV extraordinarily beat different calculations. For instance, on work F11, EBATV tracked down the worldwide least in completely run while different calculations created less fortunate outcomes for this situation and in work F9 EBATV tracked down the worldwide least in totally run and different calculations produced exceptionally less fortunate outcomes for this situation, and on work F10 and F12 EBATV tracked down the best outcome thought about on different calculations while in F10 the close to result to EBATV found by BA-DTFS in work F8 EBATV was beated by PSO,MBA and BAGW, and in work F13 EBATV was outflanked by PSO, BAT, while all calculations were outflanked in other four capacities and EBATV get the best outcome in other four capacities.

Finally why using normalize for all result in 13 functions to determine the best algorithms using the following equation:

$$Norm = \frac{max - x}{max - min}$$  \hspace{1cm} (18)

Where x is the value of objective obtained using any algorithm and max the maximum value obtained when solving function using all algorithms and min the minimum value obtained when solving function using all algorithms when the algorithm gets the norm=1 that meaning the algorithm gets the best result compared with the other algorithms and when the norm=0 that meaning the algorithm gets the worst result compared with the other algorithms. In table (3) we calculate the normalize for the average result for all function and we can found the EBATV comparison with the other algorithm ranked the first
algorithm and obtained for the best result in 11 functions from 13 and we can see that the order of the search performance of these ten algorithms is EBATV>BA-DTFS>PSO>MBA>BAGW>BAT.

In table (4) we calculate the normalize for the standard deviation result for 50 runs for all function and we can found the EBATV comparison with the other algorithm ranked the third algorithm while obtained for the best SD in 11 functions from 13 and the BA-DTFS obtained the best SD 9 only and MBA in 12 we can see that the order of the search performance of these ten algorithms is EBATV>BAGW>BA-DTFS>BAT>PSO>MBA.

4. Conclusion
To all the more likely address the UCOPs, a coordinated BA is proposed. three improved forms are considered in the combination technique. Likewise, one determination instrument with steady likelihood is utilized to change the likelihood of every procedure. To demonstrate the predominance of the calculation, the test work is utilized to contrast EBATV and different calculations. The test results show that the effectiveness of the calculation is improved. In future exploration, the exhibition of the calculation will proceed to improve and can be applied to different applications.

Future trends recommend using the multiplicity of societies with the bat algorithm, relying on some of the mechanism for dividing societies, and the possibility of using the technique of hyper-extension in directing societies and using some improvement processes to obtain highly efficient solutions.

References


Ensemble bat algorithm based on Hyper heuristic approach for solving unconstrained optimization problems

