Hybrid of Particle Swarm Optimization and Simulated Annealing for Multidimensional Function Optimization

[1]Moitree Basu, [2]Pradipta Deb, [3]Gautam Garai

[1,2]Tata Consultancy Services Limited, [3]Saha Institute of Nuclear Physics

[1]sfurti.basu@gmail.com , [2]deb.pradipta@gmail.com, [3]gautam.garai@saha.ac.in

Abstract

The classical Particle Swarm Optimization (PSO) Algorithm is very efficient and effective in solving optimization problems (both minimization and maximization). But PSO algorithm has a shortcoming of converging prematurely after getting trapped into some local optima (local optimum solution point) and considers it to be the global optima (global optimum solution point). Moreover, when we apply it to a multi-dimensional complex problem scenario, then due to some constraints it becomes nearly impossible to get out from that local optima (apparent global optima) and reach out for the global optima. Instead, all the particles starts getting converged to that apparent optimum solution. On the contrary, Simulated Annealing (SA) Algorithm can hinder the premature convergence to the local optima and diverges the particles using its strong ability of local search. Here, we propose a new hybrid algorithm of Particle Swarm Optimization (PSO) and Simulated Annealing (SA) in optimization (We applied and concentrated on minimization problems) of complex, multi-dimensional functions. The proposed algorithm is fundamentally based on the PSO algorithm, whereas, SA method is used to slow down the convergence of the swarm and to increase the swarm's probability of reaching the global optima by increasing the diversity of particles.

Keywords: Particle Swarm Optimization; Simulated Annealing; pbest; gbest; local optima; global optima.

I. Introduction

PSO optimizes a problem by having a population of particles moving around in the search-space according to velocity and location update formula to set the particle's position and velocity in each iteration depending upon the corresponding values in previous iteration. Each particle updates their position and velocity to move towards its local best position (local minima for optimization towards the minimal solution and local maxima for optimization towards the maximal solution) known and is also guided towards the best known position in the search-space by doing this iteratively, which are updated as better positions are found by other particles. If no exception occur, then the whole swarm of particles is expected to move towards the best solutions globally (global minima for optimization towards the minimal solution and global maxima for optimization towards the maximal solution) in the search space.

The Particle Swarm Optimization (PSO) algorithm is broadly used to solve continuous quantities problems and functions as well as PSO shows better performance in solving many optimization problems too. But, fundamental PSO algorithm suffers from a premature convergence syndrome that shows all particles are prone to be trapped into the local minima and cannot reach out for searching any better solution available in the search space. For this reason, the swarm end up finding a local minima as the optimal value found instead of a global minima.

Many successful experiments have been proposed to increase the diversity of the swarm particles. Some of them could also improve the convergence performance to some extent.

Here, global stochastic method such as simulated annealing (SA) [1] could be used along with PSO. SA is quite popular for its powerful feature of effective escaping from the trap of local minima. In the paper, we proposed a new solution which combines PSO algorithm with the simulated annealing algorithm (SA) and that too in such a rounding fashion, so that SA can be applied whenever PSO tends to converge into a local minima/maxima and SA diverges the solution from the apparent best . By integrating SA to PSO, the new algorithm, which we call as Hybrid PSO-SA Algorithm, we made it escape from converging into the local minimum, as well as simplify the implementation of the combined algorithm.

II. Background

A. Particle Swarm Optimization

PSO is a concept inspired by social behavioural patterns of a population or Swarms of living organisms (here we assumed them as particles). The algorithm exactly follows the behaviours observed in schools of fishes, or swarms of bees, colonies of ants, flocks of birds and human community, especially when they move from one place to another (from their nest to the source of food).

PSO algorithm is developed and inspired by the concept that particles move through the search space and their future velocities and locations are dynamically determined according to their past behaviours i.e. previous location and velocity. Therefore, the particles are prone to move towards the better search area over iterations and tend towards the optimum solution point. PSO algorithm starts with a group of random (or semi-random) particles (solution set of velocity and location of each particle) and then searches for optima by updating their location and velocity in each next loop. Here, for the sake of establishing an algorithm, each particle is treated as a volume-less particle (or just a point having location and velocity) in the n-dimensional search space. The ith particle is represented as Xi= ($xi1$, $xi2\ldots xin$).

At each iteration, the particles are updated by using following three best values:

(i) **pbest**: The best solution (fitness value) a particle (personal best) has achieved as of now is called pbest.

(ii) **lbest**: The best solution tracked by the particle swarm optimizer so far (by any particle) in the population. When a particle takes part of a population as its to neighbours, the best value is a local best and is called lbest.

(iii) **gbest**: This best value is a global best and is called gbest best of all pbest. [2]

At each iteration, existing pbest and gbest are considered and combined to calculate the location and velocity for each particle in each dimension and similarly these details will help in computing the location and velocity of the corresponding particles in the next iteration. The personal best (pbest) in the previous iteration is compared with the best solution in the neighbourhood or locality (lbest) and the location of the particle is computed according to the result of the comparison.

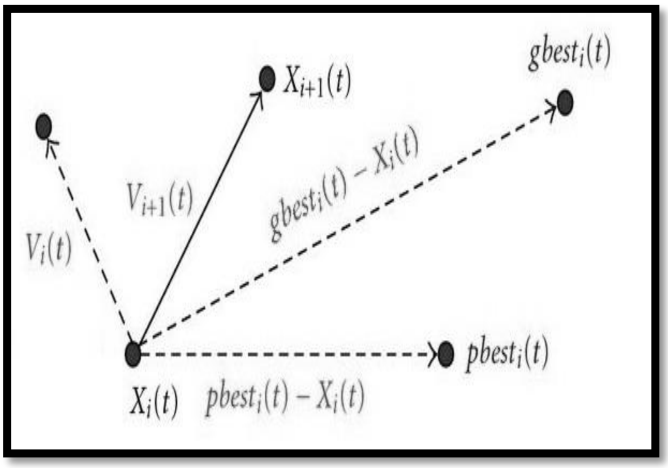


Figure 1: *Movement of each particle*

The formula to update or calculate the location of any particle (ith particle) is as follows:

$$xi+1= xi+ vi + 1 \quad (3)$$

Here an inertia factor, ω, is also introduced in the formula of calculating the velocity of any particle. The equation for velocity calculation is as follows [3]:

$$vi+1 = \omega vi + c1 * \text{random } (0, 1) * pbesti - xi + c2 * \text{random } (0, 1) * gbe \ - xi \quad (4)$$

Where random (0, 1) is a random (or semi random) number generated within the range of [0, 1]. Here, $c1$ and $c2$ are two learning factors which control the influence of the social and cognitive components (usually, $c1$= $c2$ = 2, see [4]). In (4), the value of the particle velocity (V) ranges between [Vimin; Vimax]. As, particle velocities are clamped to a certain range, it reduces the likelihood of particles leaving the search space. Note, that the values of xi are not restricted to the range [Vimin; Vimax], it only limits the maximum distance that a particle will move during one iteration.

B. Simulated Annealing Algorithm

In contrast to PSO, SA is a probabilistic approach of local search method, which can escape premature convergence at local optima. SA is based on a process in thermodynamics, where to grow a crystal, material is heated until it reaches its molten state and then, the temperature of this crystal melt is reduced very slowly, until the crystal structure is formed. In the basic SA algorithm, a random or semi-random value is considered to be the initial solution in the search space. Then the new solution value of the objective function is calculated and compared with the current solution. A move towards the new solution is made from the current solution if it provides a better result than the previous iteration. The probability of accepting a new solution is given as follows:

The calculation of this probability relies on a parameter T, which is referred to as temperature, since it plays a similar role as the temperature in the physical annealing process. To avoid getting trapped at a local minimum point, the rate of reduction should be slow. In our problem we use the following method to reduce the temperature

$$Ti + 1 = \gamma\ Ti, \text{ where } i = 0, 1\ldots \text{ and } \gamma = 0.99.$$

Thus, at the start of SA random and impractical moves may be accepted, but in the end only improving ones are likely to be allowed, which can help the procedure not to converge at a local minimum. The algorithm may be terminated after a certain volume fraction of the structure has been reached or after a pre specified iterations.

III. Hybrid PSO-SA Algorithm

This section presents a new hybrid PSO-SA algorithm which combines the advantages of both PSO (that has a strong global-search ability) and SA (that has a strong local-search ability). Other applications of hybrid PSO and SA algorithm can also be found [5, 6]. This hybrid approach makes full use of the global and local search optimization capability of both PSO and SA respectively and overcomes the weaknesses of each algorithm separately possesses. Through application of SA to PSO, the proposed algorithm is capable of escaping from a local optima and succeed in converging into the global optima in the search space.

Now, we have applied PSO and SA in a specific rounding fashion, so that they induce each other to provide fast and most optimum result for a problem. If in each iteration we'd used both PSO and SA, then SA will try to diversify the points and PSO will try to converge the points at the same time, which will in turn delay the convergence of PSO as well as the capabilities of SA will also not be effective. So, we have applied PSO in the problem in a more elaborate fashion and only applied SA when PSO has stuck at a local optima and have not updated the particle locations for a certain number of iterations, so that SA can escape from the local optima and diversifies the prematurely converged particles in the search space.

SA could also be applied in a rounded fashion after every k iterations.

The hybrid PSO-SA algorithm works as follows:

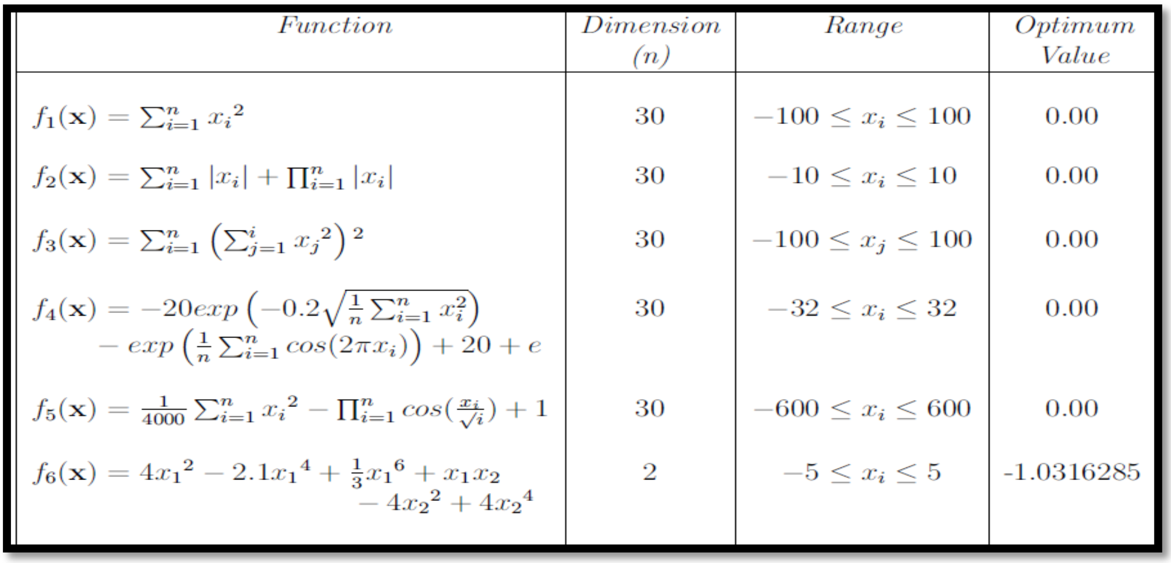
```
❖ iter ←0, cpt ← 0, Initialize swarm size particles
❖ stop criterion←maximum number of function evaluations or Optimal
  solution is not attained
❖ while Not stop criterion do
    ➢ for each particle i ← 1 to swarm size do
        ▪ Evaluate(particle(i))
        ▪ if the fitness value is better than the best fitness value (cbest) in
          history then
            • Update current value as the new cbest.
        ▪ end
    ➢ end
    ➢ Choose the particle with the best fitness value in the neighborhood
      (gbest)
    ➢ for each particle i ← 1 to swarm size do
        ▪ Update particle velocity according to Equation (3)
        ▪ Enforce velocity bounds
        ▪ Update particle position according to Equation (4)
        ▪ Enforce particle bounds
    ➢ end
    ➢ if there is no improvement of global best solution then
        ▪ cpt ← cpt + 1
    ➢ end
    ➢ Update global best solution
    ➢ cpt ← 0
    ➢ if cpt = K then
        ▪ cpt ← 0
      //Apply SA to global best solution
        ▪ iterSA ← 0, Initialize T according to Equation (5)
        ▪ current solution ← global best solution
        ▪ current cost ←Evaluate(current solution)
        ▪ while Not SA_stop_criterion do
            • while inner-loop stop criterion do
                ♦ Neighbor ←Generate(current solution)
                ♦ Neighbor_cost ← Evaluate(Neighbor)
                ♦ if Accept(current_cost, Neighbor_cost, T) then
                      o current_solution ← Neighbor
                      o current_cost ← Neighbor_cost
                ♦ end
                ♦ iterSA ← iterSA + 1
                ♦ Update (global_best_solution)
            • end
            • Update(T) according to Equation(2)
            • Update (SA_stop_criterion)
        ▪ end
    ➢ end
    ➢ iter ← iter + 1, Update (stop_criterion)
❖ end
```

IV. Experimental Result

In order to draw analogy between the performances of the proposed Hybrid PSO-SA Algorithm with the standard optimal solutions, we use benchmark functions[7,8,9,10] described in Figure 2. These functions were applied on different optimization processes and provides a trustworthy source of credible data that can be used for the purpose of optimization algorithms. For each of these functions, there could be many local optima as well as one or more global optima in their solution space. As we keep increasing the number of dimensions, the problem becomes more complex, more local optima are likely to occur and it leads to delay in converging to the correct global solution for that function. In the following experiments, we used 2, 5, 10, 15, 20 and 30 dimensional functions except in the case of Six-Hump Camel-Back Function (f6) that is two-dimensional by definition.

A. Benchmark Functions

Here we provide a chart of functions and their corresponding Optimum values for certain dimensions and range of values provided:



Function	*Dimension (n)*	*Range*	*Optimum Value*
$f_1(\mathbf{x}) = \sum_{i=1}^{n} x_i{}^2$	30	$-100 \leq x_i \leq 100$	0.00
$f_2(\mathbf{x}) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	$-10 \leq x_i \leq 10$	0.00
$f_3(\mathbf{x}) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j{}^2 \right)^2$	30	$-100 \leq x_j \leq 100$	0.00
$f_4(\mathbf{x}) = -20exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - exp\left(\frac{1}{n}\sum_{i=1}^{n} cos(2\pi x_i)\right) + 20 + e$	30	$-32 \leq x_i \leq 32$	0.00
$f_5(\mathbf{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i{}^2 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}}) + 1$	30	$-600 \leq x_i \leq 600$	0.00
$f_6(\mathbf{x}) = 4x_1{}^2 - 2.1x_1{}^4 + \frac{1}{3}x_1{}^6 + x_1x_2 - 4x_2{}^2 + 4x_2{}^4$	2	$-5 \leq x_i \leq 5$	-1.0316285

Figure 2: *Benchmark Functions used in this paper*

To verify the efficiency and effectiveness of Hybrid PSO-SA Algorithm, the experimental results of PSO-SA approach are compared with the benchmark results.

B. Comparison with Results Obtained with benchmarks

In this section we have shown the result of comparison of PSO-SA approach using benchmark functions. Here, we have taken the number of particles in the swarm as 30. The number of dimension of the searching space varies among the following values 2,5,10,15,20,30 and the number of objective function evaluations varies among these values 1000,3000,5000,7000. The results obtained after numerical simulations are shown in below Figure 3. By analysing the values of Figure (3), we conclude that the results obtained by PSO-SA algorithm are preferable in comparison with the mathematical benchmarks.

Dimension ------------------- Equation	2-D (1000 Iterations)	5-D (1000 Iterations)	10-D (1500 Iterations)	15-D (2000 Iterations)	20-D (5000 Iterations)	30-D (7000 Iterations)
Eq. 1 ($-100 \leq x \leq 100$)	0	$4.092e^{-135}$	$6.727e^{-113}$	$1.133e^{-55}$	$9.875e^{-20}$	$5.785e^{-11}$
Eq. 2 ($-10 \leq x \leq 10$)	0	$6.597e^{-59}$	$6.212e^{-24}$	$8.156e^{-11}$	$3.191e^{-08}$	$4.028e^{-07}$
Eq. 3	0	$8.949e^{-201}$	$4.086e^{-188}$	$6.891e^{-69}$	$2.115e^{-13}$	$2.637e^{-08}$

(-100 ≤ x ≤ 100)						
Eq. 4 (-32 ≤ x ≤ 32)	0	$6.567e^{-35}$	$9.439e^{-18}$	$2.074e^{-15}$	$7.665e^{-14}$	$5.306e^{-09}$
Eq. 5 (-600 ≤ x ≤ 600)	0	$3.803e^{-23}$	$6.805e^{-13}$	$7.241e^{-09}$	$5.728e^{-07}$	$6.817e^{-06}$
Eq. 6 (-5 ≤ x ≤ 5)	-1.03162845	NA	NA	NA	NA	NA

Figure 3: *Result obtained from PSO-SA Hybrid Optimization Function for the above mentioned benchmark functions*

V. Conclusion

In this paper, we have designed a hybrid algorithm (PSO-SA) that combines the individual PSO and SA algorithm in such a fashion that they overcome the shortcomings of each other but does not dominate their positive capabilities of finding optimum solution of complex functions in multi-dimensional space. Here we have shown that PSO-SA performs well in the metrics like accuracy, robustness, rate of convergence, stability and number of iterations while compared to the mentioned popular benchmark functions. We not only have discussed performance of different complex benchmark functions but have also shown the comparative performance of the proposed hybrid algorithm with respect to varying dimensions for each function. So, in future, we can try hybridize PSO with some other optimization algorithm as well as calculate and compare the result with such complex functions with dimensions more than 30 and try to achieve even more accuracy in finding optimal solution in least iterations or computational time possible.

References

[1] M. Locatelli, “Simulated annealing algorithms for continuous global optimization,” in Handbook of Global Optimization, P. M. Pardalos and H. E. Romeijn, Eds., vol. 2, Kluwer Academic, 2001, pp. 179–230.

[2] R. Poli, J. Kennedy, and T. Blackwell, “Particle swarm optimization. An overview,” in Swarm Intelligence, vol. 1, 2007, pp. 33– 57.

[3] Y. Shi and R. C. Eberhart, “A modified particle swarm optimizer,” in Proceedings of the IEEE Congress on Evolutionary Computation (CEC ’98), IEEE Computer Society, 1998, pp. 69–73.

[4] Y. Shi and R. C. Eberhart, “Empirical study of particle swarm optimization,” in Proceedings of the Congress on Evolutionary Computation (CEC ’99), 1999, pp. 1945–1950.

[5] G. Yang, D. Chen, and G. Zhou, “A new hybrid algorithm of particle swarm optimization,” in Lecture Notes in Computer Science, vol. 4115, 2006, pp. 50–60.

[6] M. Bahrepour, E. Mahdipour, R. Cheloi, and M. Yaghoobi, “Super-sapso: a new sa-based pso algorithm,” in Applications of Soft Computing, vol. 58, 2009, pp. 423–430.

[7] P. N. Suganthan, N. Hansen, J. J. Liang et al., “Problem definitions and evaluation criteria for the cec 2005 special session on real-parameter optimization,” Tech. Rep. 2005005, Nanyang Technological University, Singapore; IIT Kanpur, India, 2005.

[8] Hybrid of Particle Swarm Optimization, Simulated Annealing and Tabu Search For The Reconstruction Of Two-Dimensional Targets From Laboratory-Controlled Data

[9] Particle Swarm Optimization with Simulated Annealing for TSP, Hangzhou, 310014 CHINA.

[10] S. Yh, R. C. Eberhart, “Fuzzy Adaptive Particle SwarmOptimization, Congress on Evolutionary Computation,” 2001, pp.101-106.

Moitree Basu is currently working in Tata Consultancy Services Limited as Assistant System Engineer. She received her B.Tech. Degree in Dept. of Information Technology from Netaji Subhash Engineering College in 2012. She will be starting her PhD degree in US in fall 2014. Her current research interests include Cloud Computing, Algorithms, Computational Biology and Bioinformatics etc. She possesses interest in painting, listening to music.

Pradipta Deb is currently working in Tata Consultancy Services Limited as Assistant System Engineer. He received his B.Tech. Degree in Dept. of Information Technology from Netaji Subhash Engineering College in 2012. He will be starting his PhD degree in US in fall 2014. His current research interests include Cloud Computing, Algorithms, Computational Biology and Bioinformatics etc. He possesses interest in pencil drawing, playing musical instruments.

Gautam Garai is currently a Scientist at Saha Institute of Nuclear Physics, Kolkata, India. He joined the Institute in 1988 as Scientist 'B' and presently working as Scientist 'G'. He was an Assistant Systems Engineer at M. N. Dastur Co. Ltd., Kolkata, India till 1988 after the completion of his graduation. His undergraduate degree is B.E. in Computer Science and Engineering which he received from Bengal Engineering College, Howrah, India in 1987. His post graduate M.E. degree was also in Computer Science and Engineering from Jadavpur University, Kolkata, India. He has received Ph.D. degree in Computer Science and Engineering from Jadavpur University. He was awarded National Scholarship by Government of India in 1981 and in 1983. His research interests are in Evolutionary Algorithms, Pattern Recognition, Bio-informatics and Data mining.