Int. J. Simul. Multidisci.Des. Optim. **4**, 27-32 (2010)

DOI: 10.1051/ijsmdo/2010004

Available online at:
http://www.ijsmdo.org

Hybrid Evolutionary Optimization Algorithm MPSO-SA

Norelislam El Hami[1,2,a], Rachid Ellaia[1], Mhamed Itmi[2]

[1]Laboratory of Study and Research in Applied Mathematics (LERMA) , Mohammed V University - Engineering Mohammedia School Rabat, BP. 765 Ibn Sina avenue, Agdal, Morocco
[2]Laboratory (LITIS) of Rouen, National Institute for Applied Sciences - Rouen BP. 08, University Avenue 76801, St Etienne du Rouvray Cedex, France

Received 10 December 2009, Accepted 15 February 2010

Abstract- This paper proposes a new method for a modified particle swarm optimization algorithm (MPSO) combined with a simulated annealing algorithm (SA). MPSO is known as an efficient approach with a high performance of solving optimization problems in many research fields. It is a population intelligence algorithm inspired by social behavior simulations of bird flocking. Considerable research work on classical method PSO (Particle Swarm Optimization) has been done to improve the performance of this method. Therefore, the proposed hybrid optimization algorithms MPSO-SA use the combination of MPSO and simulated annealing SA. This method has the advantage to provide best results comparing with all heuristics methods PSO and SA. In this matter, a benchmark of eighteen well-known functions is given. These functions present different situations of finding the global minimum with gradual difficulties. Numerical results presented, in this paper, show the robustness of the MPSO-SA algorithm. Numerical comparisons with these three algorithms: Simulated Annealing, Modified Particle swarm optimization and MPSO-SA prove that the hybrid algorithm offers best results.

Key words: Global optimization; PSO; SA; evolutionary algorithm; Hybrid Methods

1 Introduction

The Particle Swarm Optimization (PSO), developed by Kennedy and Eberharts in 1995 [22], is an approximation algorithm method proposed for the optimization problem of finding the global minimum. Since then, it has been improved by many searchers. The principal of this algorithm is based on the movement of birds searching for a food in a flock; this animal behavior is simulated to the optimization research.

This method generates a group of particles, each one search for the minimum of the fitness by their own knowledge and movement, and is influenced by the search of his neighbor. If a particle finds a good site, all the others can become aware of it more or less directly, in order to take advantage of it.

Bochenek Fory have improved particle swarm optimizer in 2006 [19] by adding a new term in the velocity equation. It represents the distance between the particle position and the position of the particle neighbors' leader, the best particle among its neighbors. The complementary information provided influences swarm member behavior and, in many cases, their algorithm can improve swarm performance.

A simulated annealing algorithm SA is based on the idea in physics of annealing a solid to the state with a minimum energy. In this algorithm an initial solution is generated which is repeatedly improved by making a few alterations. This process had been already done in a constant temperature using the probability test for accepting a candidate solution which depends on the individual estimated objective function value. The algorithm is shown to converge almost surely to an optimal solution.

The modified particle swarm optimization algorithm (MPSO)-simulated annealing algorithm SA (MPSO-SA) used modified PSO as an evolutionary searching mechanism to effectively perform exploration for promising solutions within the entire region, and used SA to perform exploitation for solution improvement. Several combinations have been tested to avoid selecting an appropriate model.

Majid Bahrepour has presented, in 2009 [8], a new method called SUPER-SAPSO algorithm which fuses SA with standard PSO. The idea of this method is that the new position during the movement of the particle in the swarm is multiplied by the temperature factor and the SUPER-SAPSO assigns the temperatures locally to each particle of the swarm. The temperature is a function of error, they assign a hot temperature to particles with poorer fitness.

Behnamian and Ghomi have published, in 2010 [1], a hybrid method where SA-based local search is applied to the best solution given during the PSO search process. At high temperature, the algorithm performs exploration with certain jumping probability while at low temperature, the algorithm stresses the exploitation from the best solution.

[a]Corresponding author: norelislam@hotmail.com

Nomenclature and Abbreviations .

SH3 Shubert function with 3 dimension
SH4 Shubert function with 4 dimension
BO1 Bohachevsky1 function
BO2 Bohachevsky2 function
BR Branin function
CA Camel function
CM Cosine Mixture function
DE Dejoung function
GP Goldstein Price function
GR Griewank Price function
HAN Hansen function
HA3 Hartman function, 3 dimension
HA6 Hartman function, 6 dimension
RA Rastrigin function
RO Rosenbrock function
SK5 Shekel function
SK7 Shekel function
SK10 Shekel function
Dim Function dimension
SR Success Rate
SD Standard Deviation
EvalFNumber of Evaluated Function
Ftest Test Function
PSO Particle Swarm Optimization
SA Simulated Annealing

2 SA Algorithm

SA is a classical optimization technique that has been successfully used for solving a wide range of optimization problems. In the SA algorithm, like most of its family (Meta heuristics that are based on gradual local improvement), we started with a randomly chosen none optimal configuration and moved iteratively to another solution in the neighborhood to improve the configuration. A brief description of this algorithm is given in this section in order to show the combination with the PSO in later sections. In the first stage, the algorithm started from an initial solution S which is generated randomly, thereafter we generated a new solution S' in the neighborhood of the precedent one. For minimization problems, the new solution is accepted with certain probability P.

$$The\ probability\ P\ <\ \exp(\tfrac{\Delta E}{T})$$

where Δ E is the variation of the energy, which can also be the variation of the fitness in optimization method.
Δ E = F(S') - F(S). T is the temperature and it is considered in this study as the control parameter. This probability principle is used to select the uphill moves that may help the optimization procedure escape from local minima. In this algorithm we started from high temperature value to a lower one gradually in order to find precisely the global optimum.

Pseudocode of Simulated Annealing algorithm .

.$S \leftarrow Sinit$ Generate the initial population randomly.
.$Sbest \leftarrow S$
.$Fbest \leftarrow F(S)$
.Initialisation of T (temperature)
.Start Iterations $NbIter \leftarrow 0$
.$While\ T > Tminimum$
. $NbIter \leftarrow NbIter + 1$
. Generate S' in the neighbour of S
. $\Delta \leftarrow F(S') - F(S)$
. $if\ \Delta\ <\ 0$
. $S\ \leftarrow\ S'$
. $if F(S') < F(Sbest)$
. $Fbest \leftarrow\ F(S')$
. $Sbest \leftarrow\ S'$
. end
. $else$
. $Random\ p\ in\ [0\ 1]$
. $if\ p \leq \exp(-\Delta/T)$
. $S\ \leftarrow\ S'$
. end
. end
.end

3 MPSO Algorithm

In Particle Swarm Algorithm, each particle i is treated as a point in a space with dimension D, a position X_i, a velocity V_i and personal best position X_{besti}. The personal best position associated with a particle i is the best position that the particle has visited. The best position of all particles in the swarm is represented by the vector X_{gbest}.

$X_i = (x_{i1}, x_{i2}, ..., x_{id})$ The position of the particle.
$V_i = (v_{i1}, v_{i2}, ..., v_{id})$ The velocity of the particle.
$X_{besti} = (p_{i1}, p_{i2}, ..., p_{id})$ The best personnel position.
$X_{gbest} = (p_{g1}, p_{g2}, ..., p_{gd})$ The best global position.

$$V_{id}(t+1) = \chi(V_{id}(t) + \rho_1[X_{besti}(t) - X_i(t)] + \rho_2[X_{gbest}(t) - X_i(t)]) \quad (1)$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (2)$$

$\rho_1 = c_1 r_1$ and $\rho_2 = c_2 r_2$
$c_1\ and\ c_2$: positive acceleration components called social parameter .
χ : constriction coefficient.
$r_1\ and\ r_2$: Independent random number in the rang $[0, 1]$

We modified the velocity function by using a new term X_{Nbest} in the equation 1, which was introduced by Bochenek and Fory [19] defined as:
$X_{Nbest} = (p_{n1}, p_{n2}, ..., p_{nd})$ The best position of the neighborhood.

The equation 1 becomes :

$$V_{id}(t+1) = \chi(V_{id}(t) + \rho_1[X_{besti}(t) - X_i(t)] + \rho_2[X_{gbest}(t) - X_i(t)] + \rho_3[X_{Nbest}(t) - X_i(t)]) \quad (3)$$

$\rho_3 = c_3 r_3$
c_3 : positive acceleration components called social parameter .
r_3 : Independent random number in the rang $[0, 1]$

The initialization of the swarm and velocities is usually performed randomly in the search space, following a uniform distribution. The best positions are initially set equal to the initial swarm. After the first time increment, they will be moved by the velocity V_i in Equation (3). Then the algorithm searches for optima by updating generations. The acceleration constants c_1, c_2 and c_3 in Equation (3) represent the weighting of the stochastic acceleration terms that pull each particle towards X_{besti}, X_{Nbest} and X_{gbest} positions.c_1 represents the confidence that the particle has in itself, c_2 represents the confidence that the particle has in the swarm, and c_3 represents the confidence that the particle has in his neighbor. In most cases, the acceleration parameters c_1, c_2 and c_3 are affected to 1, however, if we want to eliminate the particle's own experience we take c_1 = 0; c_2 = 1 and c_3 = 1 or eliminate the influence of the best of the swarm we take c_1 = 1; c_2 = 0 and c_3 = 1 or we eliminate the influence of the best of the neighbor we take c_1 = 1; c_2 = 1 and c_3 = 0. Depending on the problems to resolve we can make the appropriate choices for these parameters to modify the velocity and to promote convergence.

The search procedure of a population-based algorithm such as PSO consists on the concept of neighborhood, the information regarding the best position of each neighborhood is gradually communicated to the rest of the particles through their neighbors in the ring topology, we have a neighborhoods that consist of particles belong to different partitions. In this case, particles with different behaviors can interact by sharing information through their neighborhoods. All particles in a neighbor share the same value of X_{Nbest} and each neighbor has a different value of X_{Nbest}. It is important to respect the number of particles that comprise the neighborhoods, therefore, in our experiments the swarm was divided into 7 partitions. In general there is no formal procedure to determine the optimal number or the size of the neighbor but case by case depending on the problems to resolve.

Pseudocode of Modified Particle Swarm Optimization.

Initialization (Randomly)
.$X_i \leftarrow$ Generate the initial particles of the swarm
.$V_i \leftarrow$ Generate the initial velocity of the particles
.$X_{besti} \leftarrow X_i$ Set the best positions of the particle
.$X_{gbest} \leftarrow X_i$ Set the best positions of the swarm
.$X_{Nbest} \leftarrow X_i$ Set the best positions of the neighbor
Repeat
.$For\ i = 1 :\ N$(All particles in the Swarm)
. $Fitness_i(t) \leftarrow EvaluateFitness(X_i)$
. $if\ Fitness_i(t) < Fitness_(X_{besti})(t)$
. $X_{besti} \leftarrow X_i$ particle attractor
. end
. $For\ i = 1 :\ M$(M number of neighbors in the swarm)
. $X_{Nbest} \leftarrow$ defining the best position neighbor.
. end for
. $if\ Fitness_i(t) < Fitness_(X_{gbest})(t)$
. $X_{gesti} \leftarrow X_i$ swarm attractor
. end
. $Update\ velocity\ V_{id}(t+1)$ in equation 3
. $Update\ position\ X_{id}(t+1)$ in equation 2
. $if\ X_{id}(t+1) < X_{min}$ OR $X_{id}(t+1) > X_{max}$
. $X_{id}(t+1) \leftarrow X_{random} \quad X_{random} \subseteq [X_{min} X_{max}]$
. end
.End For
Until Stop criterion

4 Proposed MPSO-SA Algorithm

In this study, a new hybrid evolutionary algorithm is proposed which incorporate the SA algorithm into a MPSO. Firstly, we have modified PSO by using three terms in the velocity equation and then we combined it with the SA algorithm to increase the diversity of the population and to improve the convergence. In the proposed algorithm called MPSO-SA, SA is used as a local search around the two best positions, the first one is the best particle in the neighborhood X_{Nbest} and the second one is the best position in the swarm X_{gbest}.

The simulation of the proposed hybrid algorithm begins with an initial population and initial temperature. The particles then randomly search according to evolutionary equations of MPSO algorithm to generate a new population, which is compared and improved by SA algorithm. Then the results obtained become the individuals of the next generation. The simulation is repeated until the terminal criterion is met, which is reached when there is no improvement of the solution.

The simulated annealing is adapted to be used in the hybridization: the number of iteration in SA is reduced to increase the performance. Also the temperature parameter T is decreased within the MPSO algorithm, in each iteration a new value of the temperature and the best positions are given to SA, which start his search around them. The mechanism starts with a high value of the temperature, so we accept a given value of the best positions, after that and during the process the temperature decreases and the search are directed towards those positions that have shown a relative advantage over others, and in the same time to guide the swarm with the probability to further increase and cover the search space.

The general procedure for the MPSO-SA algorithm can be summarized as follows:

Pseudocode of MPSO-SA .

Initialization (Rondomly)
.$X_i \leftarrow$ Generate the initial particles of the swarm
.$V_i \leftarrow$ Generate the initial velocity of the particles
.$X_{besti} \leftarrow X_i$ Set the best position of the particle
.$X_{gbest} \leftarrow X_i$ Set the best position of the swarm
.$X_{Nbest} \leftarrow X_i$ Set the best position of the neighbor
Initialisation of T (temperature)
Repeat
.$For\ i = 1:\ N$
. $Fitness_i(t) \leftarrow EvaluateFitness(X_i)$
. $if\ Fitness_i(t)\ <\ Fitness_(X_{besti})(t)$
. $X_{besti} \leftarrow X_i$ particle attractor
. end
. $if\ Fitness_i(t)\ <\ Fitness_(X_{gbest})(t)$
. $X_{gbest} \leftarrow X_i$ swarm attractor
. end
. Start SA
. $For\ i = 1:\ M$(M number of neighbors in the swarm)
. $X_{Nbest} \rightarrow$ SA algorithm
. $X_{Nbest} \leftarrow$ modify the best positions of the neighbor
. end for
. End SA
. Update velocity $V_{id}(t+1)$ in equation 3
. Update position $X_{id}(t+1)$ in equation 2
. $if\ X_{id}(t+1) < X_{min}$ OR $X_{id}(t+1) > X_{max}$
. $X_{id}(t+1) \leftarrow X_{random} \quad X_{random} \subseteq [X_{min} X_{max}]$
. end
. Update the temperature value
.end
Until Stop criterion

5 Main results of Benchmark.

To illustrate the performance of the proposed MPSO-SA, we compare it with MPSO and standard SA, some investigation on benchmark functions are reported in the following tables. These functions are standard benchmark with different sizes and are all minimization problems. In order to give right indications of relative performance for each method, 100 runs were performed in this experimental setting. The results are summarized in the following with the average values.

Table 1: Simulated annealing results

		SA			
FTest	Dim	SR	SD	Time(s)	EvalF
SH3	3	80	10.7	40	1342250.6
SH4	4	20	6.76e+003	69	2074982.6
GR	50	10	17.0684	9.6e+002	2814728.3
RO	10	23	44.9	78	2796580.5
HAN	2	100	0.0308	50	1945584.2
SK5	4	60	1.7	2.1e+002	2205679.4
SK7	4	70	0.436	2.8e+002	2195135
SK10	4	100	0.00227	3.6e+002	2104962.2
HA3	3	100	9.02e-005	85	2440969
HA6	6	20	0.00684	5.6e+002	13014268.2
DE	3	100	6.59e-006	47	2463189
CA	2	100	7.45e-007	52	2463189
GP	2	100	9.49e-006	51	2272379.8
CM	4	50	0.000296	55	2438221.8
BR	2	100	4.53e-007	49	2463189
BO1	2	100	6.65e-007	49	2463189
BO2	2	100	4.06e-006	49	2463189
RA	2	30	0.00447	48	2305790.6

Table 2: Modified Particle Swarm Optimisation results

		MPSO			
FTest	Dim	SR	SD	Time(s)	EvalF
SH3	3	100	3.58e-012	3.6	229284.02
SH4	4	100	1.05e-011	4.6	269959.4
GR	50	40	0.446	20	582084.6
RO	10	62	1.64	7.1	476140
HAN	2	100	1.27e-014	3.1	241828.2
SK5	4	50	3.13	7.7	188575.6
SK7	4	70	3.2	8.9	170397.6
SK10	4	70	3.48	10	139824.58
HA3	3	100	6.49e-015	4.6	248340.8
HA6	6	51	0.0598	5.5	239534.8
DE	3	100	0	5.4	476140
CA	2	100	9.93e-017	2	156030
GP	2	100	5.58e-015	2.6	203344.4
CM	4	100	9.02e-015	3.1	246113
BR	2	100	0	1.2	99409.8
BO1	2	100	0	0.71	61088
BO2	2	100	0	0.72	63182.4
RA	2	100	0	0.83	69802.6

Table 3: Proposed MPSO-SA results

		MPSO-SA			
FTest	Dim	SR	SD	Time(s)	EvalF
SH3	3	100	3.69e-012	3.4	137372.08
SH4	4	100	7.4e-011	4.3	455889.6
GR	50	100	1.6e-010	18	570334.2
RO	10	91	1.02	4.9	243578
HAN	2	100	2.6e-013	3.1	364928
SK5	4	58	5.7	6.9	256903
SK7	4	82	3.25	9.3	198789
SK10	4	83	5.76	8.6	176896
HA3	3	100	6.55e-015	2.9	80846.1
HA6	6	61	0.132	3.4	76704.18
DE	3	100	0	3.4	213560
CA	2	100	1.57e-016	1.9	83971.2
GP	2	100	1.06e-014	2	86159.2
CM	4	100	1.16e-014	0.74	32718.4
BR	2	100	0	1.2	57040
BO1	2	100	0	0.73	34725.9
BO2	2	100	0	0.74	35043.9
RA	2	100	0	0.61	28264

The results show, in most of the time, that the MPSO-SA can achieve the optimal solution with higher probability and the computation time is lower than the other methods SA and MPSO. The SA can guarantee the optimum solution when the problem size is small, but it will take a long time when the problem size is larger. The MPSO can solve the larger problem in an acceptable time, which demonstrates the powerful explore-ability of the MPSO algorithm. We can mention that sometimes it cannot find the optimal solution. The results reveal that the number of function evaluations of MPSO-SA is less than the other algorithms. The comparison strongly suggests that the MPSO-SA is able to efficiently improve the computational performance of complex optimization problems.

The performance of the algorithm is evaluated in comparison with results obtained from other algorithms for a number of benchmark instances. The new algorithm is very effective and efficient. It can find the optima for most test instances and running time is less than almost all other algorithms. Because of the generality of MPSO-SA, it can be applied to many optimization problems. These results indicate that the proposed algorithm is an attractive alternative for solving optimization problems.

6 Conclusion

In this paper, we presented the hybridization of population-based evolutionary searching ability of our new method MPSO-SA. The results of effective hybrid MPSO-SA was proposed. The results of simulation indicated that the disadvantage of classical PSO is conquered by MPSO method. The ability of global optimality is toned up by MPSO-SA. The balance between the global exploration and the local exploitation was stressed. On one hand, MPSO-SA applied the evolutionary searching mechanism of MPSO which is modified from the classical one and characterized by individual improvement plus population cooperation and competition to effectively perform exploration. On the other hand, MPSO-SA used an adaptive local search by SA method to perform exploitation. The simulation results and comparisons with the three approaches demonstrated the superiority of the proposed MPSO-SA in terms of searching quality, robustness and the low number of function evaluation. Our proposed method is a promising solution and encourages us to apply it to other Optimization problems.

References

1. J.Behnamian, S.M.T Fatemi Ghomi, *Development of a PSO-SA hybrid metaheuristic for a new comprehensive regression model to time-series forecasting*, Expert Systems with Applications, 974-984, (2010).

2. V. Savsani, R.V. Rao, D.P. Vakharia, *Optimal weight design of a gear train using particle swarm optimization and simulated annealing algorithms*, Mechanism and Machine Theory, 531-541, (2010).

3. M.M. Ali, M.N. Gabere, *A simulated annealing driven multi-start algorithm for bound constrained global optimization*, Journal of Computational and Applied Mathematics, 2661-2674, (2010).

4. T. Niknam, B. Amiri, J. Olamaei, A. Arefi, *An efficient hybrid evolutionary optimization algorithm based on PSO and SA for clustering*, Journal of Zhejiang University SCIENCE, 512-519, 2009.

5. S. Sitarz, *Ant algorithms and simulated annealing for multicriteria dynamic programming*, Computers & Operations Research, 433-441, (2009).

6. Y. Zhaoa, W. Zub, H. Zeng, *A modified particle swarm optimization via particle visual modeling analysis*, Computers and Mathematics with Applications, 2022-2029, (2009).

7. M.H. Alrefaei, A.H. Diabat, *A simulated annealing technique for multi-objective simulation optimization*, Applied Mathematics and Computation, 3029-3035, (2009).

8. M. Bahrepour, E. Mahdipour, R. Cheloi, M. Yaghoobi, *SUPER SAPSO: A New SA-Based PSO Algorithm*, Applications of Soft Computing, 423-430, (2009).

9. W. Du, B. Li, *Multi-strategy ensemble particle swarm optimization for dynamic optimization*, Information Sciences, 3096-3109, (2008).

10. L. Lamberti, *An efficient simulated annealing algorithm for design optimization of truss structures*, Computers and Structures, 1936-1953, (2008).

11. S.W. Lin, T.Y. Tseng, S.Y. Chou, S.C. Chen, *A simulated-annealing-based approach for simultaneous parameter optimization and feature selection of back-propagation networks*, Expert Systems with Applications, 1491-1499, (2008).

12. L.L. Li, D.H. Zhou, L. Wang, *Fault Diagnosis of Nonlinear Systems Based on Hybrid PSOSA Optimization Algorithm*, International Journal of Automation and Computing, 183-188, April (2007).

13. B. Liu, L. Wang, Y.H. Jin, *An effective hybrid particle swarm optimization for no wait flow shop scheduling*, Int J Adv Manuf Technol, 1001-1011, (2007).

14. F. Zhao, Y. Hong, D. Yu, Y. Yang, Q. Zhang, H. Yi, *A hybrid algorithm based on particle swarm optimization and simulated annealing to holon task allocation for holonic manufacturing system*, Int J Adv Manuf Technol, 1021-1032, (2007).

15. P.S. Shelokar, P. Siarry, V.K. Jayaraman, B.D. Kulkarni, *Particle swarm and ant colony algorithms hybridized for improved continuous optimization*, Applied Mathematics and Computation, 129-142, (2007).

16. R. Brits, A.P. Engelbrecht, F. van den Bergh, *Locating multiple optima using particle swarm optimization*, Applied Mathematics and Computation, 1859-1883, (2007).

17. Y. Jiang, T. Hu, C. Huang, X. Wu, *An improved particle swarm optimization algorithm*, Applied Mathematics and Computation, 231-239, (2007).

18. Y. Liu, Z. Qin, Z. Shi, J. Lu, *Center particle swarm optimization*, Neurocomputing, 672-679, (2007).

19. B. Bochenek, P. Foryś, *Structural optimization for post buckling behavior using particle swarms*, Struct Multidisc Optim, 521-531, (2006).

20. D. Chaojin, Q. Zulian, *Particle Swarm Optimization Algorithm Based on the Idea of Simulated Annealing*, International Journal of Computer Science and Network Security, **6**(10), October (2006).

21. W.J. Xia, Z.M. Wu, *A hybrid particle swarm optimization approach for the job shop scheduling problem*, Int J Adv Manuf Technol, 360-366, (2006).

22. R.C. Eberhart, J. Kennedy, *A new optimizer using particle swarm theory*, in: Proceedings Sixth Symposium on Micro Machine and Human Science, IEEE Service Center, Piscataway, NJ, 39-43, (1995).