International Journal of Computer Science and Network (IJCSN)
Volume 1, Issue 1, February 2012 www.ijcsn.org ISSN 2277-5420

Performance Evaluation of Dynamic Particle Swarm Optimization

[1]Ms. Hemlata S. Urade, [2]Prof. Rahila Patel

[1] Department Computer Science & Engineering, RCERT,RTMNU
Chandrapur, Maharashtra, India

[2] Department Computer Science & Engineering, RCERT, RTMNU
Chandrapur, Maharashtra, India

Abstract
In this paper the concept of dynamic particle swarm optimization is introduced. The dynamic PSO is different from the existing PSO's and some local version of PSO in terms of swarm size and topology. Experiment conducted for benchmark functions of single objective optimization problem, which shows the better performance rather the basic PSO. The paper also contains the comparative analysis for Simple PSO and Dynamic PSO which shows the better result for dynamic PSO rather than simple PSO.

Keywords: *Dynamic PSO, Multiobjective Optimization, Optimization, PSO*

1. Introduction

Optimization has been an active area of research for several decades. As many real-world optimization problems become increasingly complex, better optimization algorithms are always needed. Unconstrained optimization problems can be formulated as a D-dimensional minimization problem as follows:
Min f (x) x=[x1+x2+……..xD]
where D is the number of the parameters to be optimized.
subjected to: Gi(x) <=0, i=1…q
Hj(x) =0, j=q+1,……m
Xε [Xmin, Xmax]D, *q* is the number of inequality constraints and *m-q* is the number of equality constraints.
The particle swarm optimizer (PSO) is a relatively new technique. Particle swarm optimizer (PSO), introduced by Kennedy and Eberhart in 1995, [1] emulates flocking behavior of birds to solve the optimization problems. In PSO, each solution is regarded as a particle. All particles have fitness values and velocities.

During an iteration of the PSO, each particle accelerates independently in the direction of its own personal best solution found so far, as well as the. Direction of the global best solution discovered so far by any other particle. Therefore, if a particle finds a promising new solution, all other particles will move closer to it, exploring the solution space more thoroughly. Typical implementations of PSO start with a reasonably sized swarm (about 40 particles). These particles are initialized with a random distribution within the solution space. As the iterations proceed, the particles will tend to cluster towards a global optimum.

In each iteration, the fitness function is evaluated to find the optimality of each proposed solution (particle), and then the location of each particle is updated to drive towards convergence. Typically, the optimization process is repeated about minimum 10,000 times to allow the particles to converge on the global optimum. If the fitness function is complex, then the per-iteration evaluation and update process will tend to be long. After a while, the particles tend to converge and repeating the evaluation for all particles per iteration will not add substantial improvement. Most particles would have converged on extremely close locations that there is no need to repeat the evaluation and update for each and every one of them.

In this paper we have conducted the experimental performance on some benchmark function with the dynamic PSO. The simple PSO is considered as fixed

swarm size and fixed topological environment. We perform this simulation work with different swarm size in each iteration of PSO and also having variation in topology. The rest of the paper organized as follow: section 2 represents related work regarding Dynamic PSO, section 3 describe basic of dynamic particle swarm optimization, section 4 describe the experimental performance which is carried out for standard benchmark function, and finally section 5 focus on our conclusion and future scope.

2. Related Work

Swarm Intelligence (SI) is an innovative distributed intelligent paradigm for solving optimization problems that originally took its inspiration from the biological examples by swarming, flocking and herding phenomena in vertebrates. Ant colony optimization, Genetic algorithm, particle swarm optimization are various evolutionary algorithms proposed by researchers. Due to simplicity in PSO equation and fast convergence, PSO is found to be best among these. After analyzing each parameter in PSO equation Yuhui Shi and Russell Eberhart [2] proposes new parameter "w" called inertia weight in the basic equation. It can be imagined that the search process for PSO without the first part is a process where the search space statistically shrinks through the generations. It resembles a local search algorithm. Addition of this new parameter causes exploration and exploitation in search space. Firstly value of w was kept static. Later on it was kept linear from 0.9 to 0.4.

Eberhart, Russell and Shi Yuhui [3] in 2000 compare these results with constriction factor. A small change in equation has been made. When Clerc's constriction method is used, φ is set to 4 and constant multiplier is thus set to 0.729. According to Clerc, addition of constriction factor may be necessary to insure convergence of particle swarm optimization algorithm The PSO algorithm with the constriction factor can be considered as a special case of the algorithm with inertia weight.

F. van den Bergh, A. P.E ngelbrecht invented Guaranteed Convergence Particle Swarm Optimizer (GCPSO) [4]. The GCPSO has strong local convergence properties than original PSO. This algorithm performs much better with the small number of particle. The new phenomenon is defined called as stagnation by these GCPSO i.e., if a particle's current position coincides with the global best position particle, then the particle will only move away from the point if its previous velocity and w are non-zero. If their previous velocities are very close to zero, then all the particles will stop moving once they catch up with the global best particle, which may lead to premature convergence of the algorithm. In fact, this does not even guarantee that the algorithm has converged on a local minimum it merely means that all the particles have converged on the best position discovered so far by the swarm.

The original PSO is easily fall into local optima in many optimization problems. The problem of premature convergence is solved by the OPSO. It allows OPSO [5] to continue search for global optima by applying opposition based learning. The OPSO use the concept of Cauchy mutation operator. The OPSO based on opposition- based learning method. The OBL method has been given by Hamid R.Tizhoosh, it is explained as when evaluating a solution x to a given problem, we can guess the opposite solution of x to get better solution of x'. By doing this the distance from the optima solution can reduce. The opposite solution x can be calculated as x`= a + b - x where x. R within [a, b].

Hierarchical PSO [6] is known as hierarchical version of PSO called as H-PSO. In this algorithm the particles are arranged in a dynamic hierarchy. In H-PSO, a particle is influenced by its own so far best position and by the best position of the particle that is directly above it in the hierarchy. In H-PSO, all particles are arranged in a tree that forms the hierarchy so that each node of the tree contains exactly one particle. If a particle at a child node as found a solution that is better than the best so far solution of particle at parent node, then these two particles are exchanged. In this algorithm the topology used as regular tree in which hierarchy is defined in terms of height and branching degree. This hierarchy gives the particles different influence on the rest of the swarm with respect to their fitness. Each particle is neighbored to itself and its parent in the hierarchy. Only the inner nodes on the deepest level might have a smaller number of children so that the maximum difference between the numbers of children of inner nodes on the deepest level is at most one. In order to give the best individuals in the

swarm a high influence, particles move up and down the hierarchy.

Mendes [7] proposed another efficient approach which deals with stagnation is the fully informed particle swarm optimization algorithm (FIPSO) [7]. FIPSO use best of neighborhood velocity update strategy. In FIPSO each particle uses the information from all its neighbors to update its velocity. The structure of the population topology has, therefore, a critical impact on the behavior of the algorithm which in turn affects its performance as an optimizer. It has been argued that this happens because the simultaneous influence of all the particles in the swarm "confounds" the particle that is updating its velocity, provoking a random behavior of the particle swarm.
New Particle Swarm Optimization(NPSO)[8] is the idea of NPSO came from our personal based experience that an individual not only learns from his or her own and other individuals' previous best, but also learns from his or her own and other individuals' mistakes. Each particle tries to leave its previous worst position and its group's previous worst position. NPSO has the better solution result than the originally PSO as more seeds are needed when the search dimension gets larger and other parameters might need to change too. The solutions might get out of local minima with more random numbers generated. Position change limits may be tuned too rather than original PSO.

Eberhart proposed a discrete binary version of PSO for binary problems [9]. In their model a particle will decide on "yes" or " no", "true" or "false", "include" or "not to include" etc. also this binary values .In binary PSO, each particle represents its position in binary values which are 0 or 1. Each particle's value can then be changed (o r better say mutate) from one to zero or vice versa. In binary PSO the velocity of a particle defined as the probability that a particle might change its state to one. The novel binary PSO [9] solve the difficulties those are occurred in binary PSO. In this algorithm the velocity of a particle is its probability to change its state from its previous state to its complement value, rather than the probability of change to 1.

In the PSO world, there exist global and local PSO versions. Instead of learning from the personal best and the best position achieved so far by the whole population, in the local version of PSO, each particle's velocity is adjusted according to its personal best and the best performance achieved so far within its neighborhood. Kennedy claimed that PSO with large neighborhood would perform better for simple problems and PSO with small neighborhoods might perform better on complex problems [10]. Kennedy and Medes discussed the effects of different neighborhood topological structures on the local version PSO [11]. Suganthan applied a combined version of PSO where a local version PSO is run first followed by a global version of PSO at the end [12].

Hu and Eberhart proposed a dynamically adjusted neighborhood when they solve the multi-objective optimization problems using PSO [13]. In their dynamically adjusted neighborhood, for each particle, the *m* closest particles are selected to be its new neighborhood. Veeramachaneni and his group developed a new version of PSO; Fitness-Distance-Ratio based PSO (FDR-PSO), with near neighbor interactions [14]. When updating each velocity dimension, the FDR_PSO algorithm selects one other particle, nbest, which has higher fitness value and near the particle being updated, in the velocity updating equation. In Mendes and Kennedy's fully informed particle swarm optimization algorithm, all the neighbors of a particle are weighted and used to calculate the velocity [15].

3. Dynamic Particle Swarm Optimization

While searching for food, the birds are either scattered or go together before they locate the place where they can find the food. While the birds are searching for food from one place to another, there is always a bird that can smell the food very well, that is, the bird is perceptible of the place where the food can be found, having the better food resource information. Because they are transmitting the information, especially the good information at any time while searching the food from one place to another, conducted by the good information, the birds will eventually flock to the place where food can be found. As far as particle swam optimization algorithm is concerned, solution swam is compared to the bird swarm, the birds' moving from one place to another is equal to the development of the solution swarm, good information is equal to the most optimist solution, and the food resource is equal to the most optimist solution during the whole course.

Particle swarm optimization is a global optimization algorithm for dealing with problems in which a best solution can be represented as a point or surface in an n-dimensional space. Hypothesis are plotted in this space and seeded with an initial velocity, as well as communication channel between the particles. Particles then move through the solution space and are evaluated according to some fitness function after each timestamp. Over time particles are accelerated towards those particles within their grouping which have better fitness values. In dynamic PSO there is variation with swarm size and variation in topology.

The dynamic particle swarm optimization concept consists of, at each time step, changing the velocity of (accelerating) each particle toward its pbest and lbest (for lbest version). Acceleration is weighted by random term, with separate random numbers being generated for acceleration towards pbest and lbest locations. After finding the best values, the particle updates its velocity and positions with following equations.

V[id]=v[id]+c1*r(id)*(pbest[id]-
x[id])+c2*r*(id)(gbest[id]-x[id])------ (1)

x[id] = x[id]+v[id]----------(2)

where,
v[id] is particle velocity
x[id] is the current particle
r (id) is random number between (0, 1)
c1 and c2 are learning factors usually c1=c2=2.

3.1 The Pseudo Code of Dynamic PSO

For each particle
Initialize Function value
END
Calculate average fitness value
Do
For each particle
If fitness value is less than average
Consider the particle
Calculate fitness value.
If the fitness value is better than the best Fitness value (pbest) in history
Set current value as the new pbest.
END
Choose the particle with the best fitness value of all the particles as the gbest.
For each particle
Calculate particle velocity according to equation (1)
Update particle position according equation (2)
END
While maximum iterations or minimum error criteria is not attained

4. Experimental Work

Here our aim is to obtain the better optimized values with the help of this dynamic PSO algorithm. Dynamic PSO can be defined as varying characteristics of PSO while experimentation is running. Characteristic include topology, swarm size, search space. If topology or swarm size can be change during process, then it is treated as dynamic particle swarm optimization. So we carry out this simulation work on some standards benchmark function. And finally we compare the results with the result obtained by basic PSO. And we get the better result than Basic PSO. Thus we can get more optimized value by using Dynamic PSO rather than the PSO.

We perform simulation to study particle behavior for dynamic PSO and Simple PSO in 2 and 10 dimensional spaces. Values of parameter involved in equation are considered as 0.7 for inertia, 1.49 for c1 and c2 and particle swarm size 40 for first iteration. Simulation has been carried out for 100000 iterations. Experiment has been conducted for standard benchmark functions. Following are results found during experimentation. The comparative result analysis is as follows: - for e.g., for Rastrigin benchmark function we got 0 and 0.0019 for 2 dimensions by using dynamic PSO whereas the simple PSO gives 9.0428e-005 and 0.0022. Further results are shown in following table.

Table 1: Comparative Result Analysis for Dynamic PSO and Simple PSO

	Formula	Dimensions	Dynamic PSO	Simple PSO
Rastrigin	$f(x)=\sum_{i=1}^{D}(x_i^2-10\cos(2\pi x_i)+10)$ where $x\in[-5.12,5.12]^D$	2	0	9.0428e-005
		10	0.0019	0.0022
Griewank	$f(x)=\sum_{i=1}^{D}\frac{x_i^2}{4000}-\prod_{i=1}^{D}\cos(\frac{x_i}{\sqrt{i}})+1$ where $x\in[-600,600]^D$	2	0	9.8608e-005
		10	0	0.0018
Ackley	$f(\vec{x})=20+e-20e^{-0.2\sqrt{\frac{\sum_{i=1}^{n}x_i^2}{n}}}-e^{\frac{\sum_{i=1}^{n}\cos(2\pi x_i)}{n}}$ $-30\le x_i\le 30$	2	1	1.101
		10	1.7764e-020	3.4235e-004
Sphere	$f(x)=\sum_{i=1}^{D}x_i^2$ where $x\in[-5.12,5.12]^D$	2	0.02	0.231
		10	0.2761	0.1913
Rosenbrock	$f(x)=\sum_{i=1}^{D-1}[100(x_i^2-x_{i+1})^2+(x_i-1)^2]$ where $x\in[-2.048,2.048]^D$	2	0	0.022
		10	0.00722	0.0144

5. Conclusion & Future Scope

A dynamic particle swarm optimization is introduced in this paper. From the simulation result we can conclude that the dynamic PSO having better optimizing performance rather than simple PSO. This gives us the future direction for solving multiobjective optimization problems by the dynamic PSO. It can be assumed that dynamic PSO is better to solve multiobjective optimization. So future direction is to apply dynamic PSO on multiobjective optimization.

References

[1] James Kennedy and Russel Eberhart," Particle Swarm Intelligence", In Washington DC 1995.
[2] Yuhui Shi and Russell Eberhart," A Modified Particle Swarm Optimizer", IEEE 1998.
[3] Xiang-Han Chen, Wei-Ping Lee, Chen yie Liao, Jag-Ting Dai,"Adaptive Constriction Factor for Location-related Particle Swarm", Proceedings of the 8th WSEAS International Conference on Evolutionary Computin,Vancouver, British Columbia, Canada, June 19-21, 2007.
[4] F. Vanden Bergh, A. P.E. ngelbrecht "A New Locally Convergent Particle Swarm Optimizers" IEEE 2010.
[5] Hui Wang, Youg Lie, Sanyou Zeng, Hui Li," Opposition based particle swarm algorithm with Cauchy Mutation" 2007.
[6] Stefan Janson and Martin Middendorf "A hierarchical particle swarm optimizer and its Adaptive variants" IEEE 2007.Macro A.
[7] Montes de Oca and Thomas Stutzle," Fully Informed Particle Swarm Optimization," IEEE 2007.
[8] Chunming Yang and Dan Simon, "A New Particle Swarm Optimization Technique" IEEE 2010.
[9] Mjtavi Ahmadieh Kinanesar, A Novel Binary Particle Swarm Optimization" IEEE 2007.
[10] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance ". Proc. of IEEE Congress on Evolutionary Computation (CEC 1999), Piscataway, NJ. pp. 1931-1938, 1999.
[11] J. Kennedy and R. Mendes, "Population structure and particle swarm performance ". Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2002), Honolulu, Hawaii USA. 2002.
[12] P. N. Suganthan, "Particle swarm optimizer with neighborhood operator," Proc. of the IEEE Congress on Evolutionary Computation (CEC 1999), Piscataway, NJ, pp. 1958-1962, 1999.
[13] X. Hu and R. C. Eberhart, "Multiobjective optimization using dynamic neighborhood particle swarm optimization," Proc. IEEE Congress on Evolutionary Computation (CEC 2002), Hawaii, pp. 1677-1681, 2002.
[14] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," Proc. IEEE Swarm Intelligence System,Indianapolis, Indiana, USA, pp. 174-181, 2003.
[15] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: simpler, maybe better". IEEE Transactions on Evolutionary Computation, 8(3):204 - 210, June 2004.

.