Particle Swarm Optimization and Crossover

Olapeju Latifat Ayoola

School of Computer Science & Informatics
University College Dublin

Abstract

This paper describes evolutionary optimization algorithm that is based on particle swarm but with addition of crossover which is one of the evolution operators in genetic algorithm. A comparison between "*Standard Particle Swarm Optimization (PSO)*" and *"PSO that includes Crossover"* will be made. PSO is known to find global optima for a given solution; meanwhile *GA is also a search heuristic that finds solution to a given problem*. Experiments were carried out on a PSOpt program [6] to observe whether Selection and Crossover improve PSO in finding global optima. Discovery is made that for some problems *"Standard PSO*" works better while in other problem *"PSO and Crossover"* improve PSO in finding global optima.

1. Introduction

Particle Swarm Intelligence (PSO) is a type of "Swarm Intelligence", whereby a group of particles interacts with one another to find a specific goal. PSO was established and built up by Dr. Eberhart and Dr. Kennedy. [2, 3, 4, 5]

Though *PSO* is similar to the evolutionary computation technique such as "*Genetic Algorithm (GA)", PSO* doesn't have evolution operators such as *mutation* and *crossover* like *GA*. This paper describes the experiment performed to implement *"Crossover*" in "*PSO*" and the outcome of the experiment in finding global optima.

A group of particles search around a space for a particular solution to a given problem by communicating with one another whenever they reach a location where they find some solution (*pbest i.e., the personal best from personal learning*). The location of best value found by any of the particles in the group is the local best (*lbest*) and the best solution they found so far is the *gbest*. They progress in their search until the best solution (i.e., Global best) is found. The *gbest* could be same as the global best and that occurs when the best possible solution has been found in the search space.

During the search the swarms dislocate from their current locations by applying a velocity vector to the locations at which they currently found a solution. The size and direction of velocity in which the swarm used to dislocate from their current location, is influenced by the swarm's adaptation traits in that current location. These adaptation traits are the swarm's personal learning (*pbest)*, peer-learning (*gbest*) and the momentum. The equation for the velocity in which the swarm uses when updating location is:

$$\mathbf{V}i\ (t+1) = w\ \mathbf{V}i\ (t) + c1\ r1\ (\mathbf{Y}i - \mathbf{X}i\ (t)) + c2\ r2(\mathbf{Y}^* - \mathbf{X}i\ (t)) \qquad (1)$$

Where $\mathbf{V}i$ is the velocity vector, w is the momentum weight, $c1$ and $c2$ are the positive constants, $r1$ and $r2$ stands for the random number drawn from $U\ (0, 1)$, $\mathbf{X}i$ is the swarm position in the optimization history, $\mathbf{Y}i$ stands for the personal best of the *ith* particle (i.e., the particle's personal history), and $\mathbf{Y}^*$ represents the peer influenced global best position of a swarm. $\mathbf{X}i$ is used to determine and update position of swarms. *W* is beneficial to the performance of PSO, it was introduced by Eberhart and Shi [8].

The maximum velocity set by a user can affect the performance of the PSO. The parameter for velocity can neither be too little nor too much. If it's too little, swarm will converge on the first best solutions while if it's too much the swarm many never converge. Random number elements r1and r2 in the equation (1) affect the performance of the optimization by influencing the outcome of the optimization.

1.1 The Concept of PSO Experiment

PSOpt program [6] initialized a population of random solutions; it searches for optima through particles interaction and adaptation. These particles represent a potential solution to the objective they are searching for. They kept searching for better optima in comparison to the previous optima found; they used their personal and social "learning rate" in accelerating towards "the location of the global best point" so far. [1]

The acceleration and deceleration of a particle was set to be between 0.9 and 0.4 for it not to converge rapidly. To maintain diversity, the time that a specific particle has a solution and tries to communicate with it's neighbour and pass it's personal learning, has to been identified and managed.

Selection and Crossover will be implemented to see if there is improvement search for the global optima using the same parameters that are specified above.

2. Experiments

Experiments were performed in two programs; these two programs optimized four different two-dimensional objective functions. These functions basically are Parabola function, Eggbox (2) function, Circle (1&2) functions shown in the figures below. These functions are basically different search spaces. The two programs optimized are Standard PSO [6] and a modified Standard PSO that includes Tournament Selection and One-point Crossover.

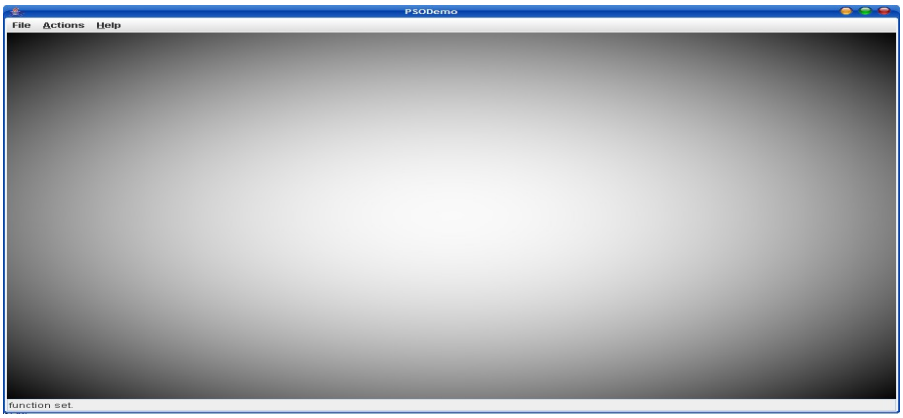

Fig.2.0 The Parabola Function

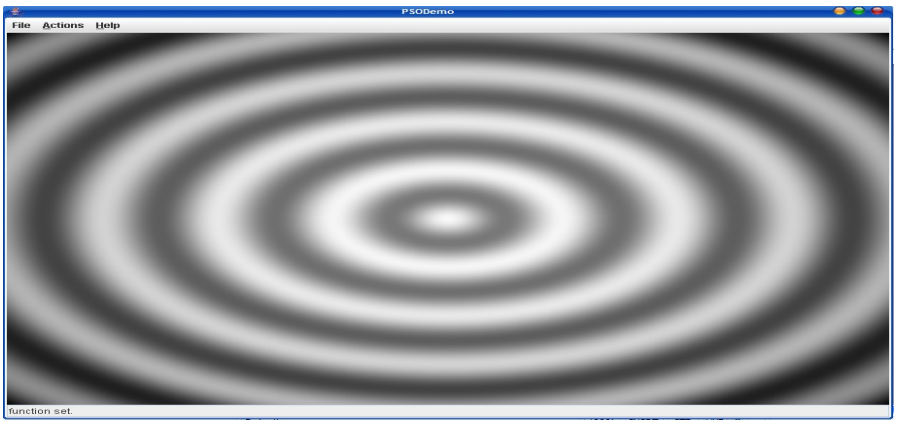

Fig.2.1 The Circle1 Function

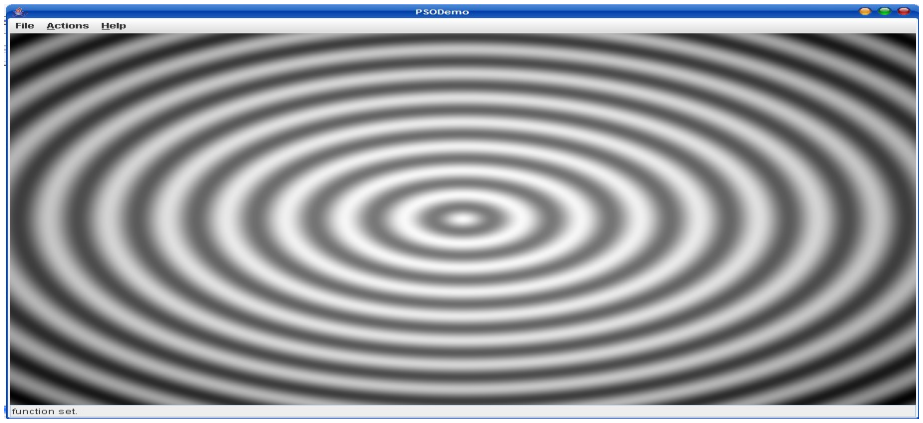

Fig.2.2 The Circle2 Function

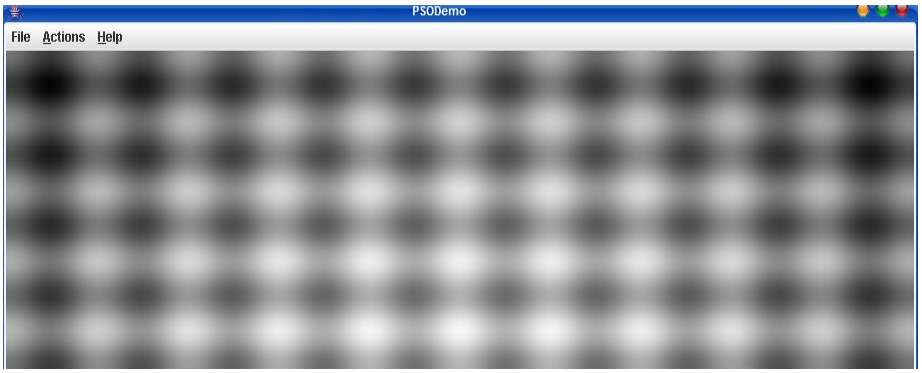

Fig.2.3 The Eggbox2 Function

In both programs, a population of 100 is generated, the acceleration factor is set to 0.9, deceleration factor is set 0.4, and the deceleration decay is set to 0.3. The tail length which is the personal history of the swarm is set to 2. Epochs of 1,000 and delay of 100 between Epochs was chosen. The Epoch is a stage whereby a distinct solution is found. It is used to manage the communication rate of particle when a *pbest* is discovered. The more epochs (i.e., solution found) there is to compute, the more, the swarm locations are continued to be updated but once iteration reaches 1000, the search halts.

Java programming language random number generator- java.util.Random was used to generate random number to improve performance of PSO.

2.1 Standard Particle Swarm Optimization

In PSOpt, a function is chosen to be optimized. PSOpt program [6] creates a swarm of particles, and start optimization. During optimization, particles fitness functions are calculated, the swarm locations are updated to the next best solution by continuous review and computing of particle's locations, then updates of the best local solution are retrieved and the *gbest*, which is the best optima found so far, is updated.

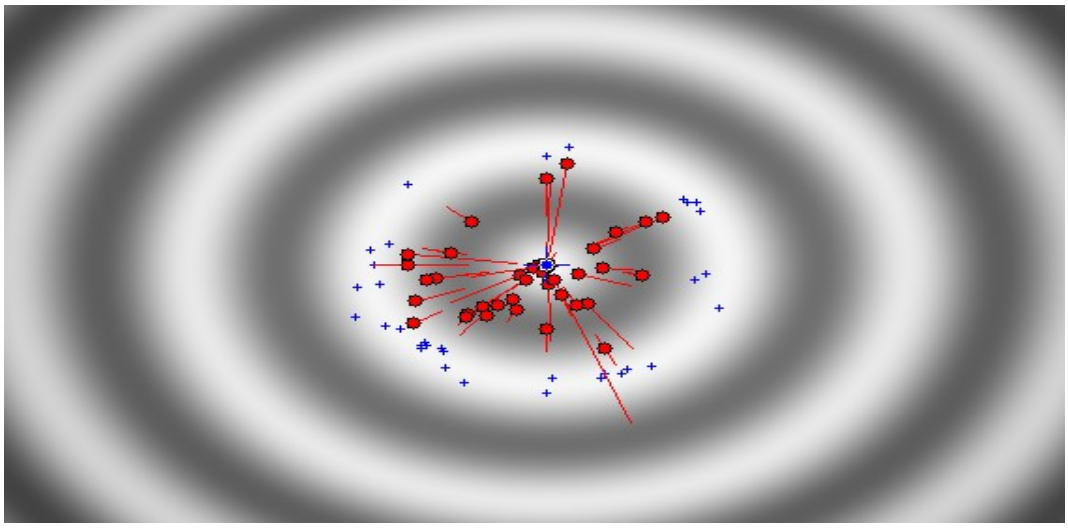

Fig.2.4 Circle 2 Function Optimized By 100 Particles

The functions used in these experiments are visualized by level of grey, for example, see the **Fig.2.4** above. The "Circle 2" Function shows particles (shown in red circles) and the swarms moving towards the global best in red lines (i.e., the swarm's trajectory). The lighter the color of the function, the function value becomes higher.

The global best is shown as a small blue cross-hair, which is located in centre (see **Fig.2.4)**, the local best are shown as small blue crosses which are shown in **Fig.2.4** above.

2.2 Particle Swarm Optimization and Crossover

In PSOpt program [6], "Tournament Selection" and Crossover was implemented. When the current locations of particles is about to be updated, two particles are randomly picked, their fitness is compared, the best fitness becomes the first parent. The same process is performed to get the second parent.

After the two best fitted parents are picked, the current location in the search space and velocity associated with the location, are retrieved and converted to *Long* type. The current location and velocity are the two parent attributes. These attributes are converted into *String* types, and then the first parent attributes are combined, the same process is done for the second parent. From these concatenated attributes crossover point is randomly picked in first parent, the same crossover point is chosen for the second parent. The *String* on the right-hand side of the crossover site is swapped between both parents. The attributes are then split back to the current location and velocity coordinates. The separated current locations and velocities are then converted back from *String* into *Long,* and then the *Long* values are converted back to the original type which was *double*. Two children then replaced the parent particles in the swarm.

The swarm locations (i.e., particles' current position in the space that hold solutions) are checked, the found locations are updated, the best local solution are computed and update, the global best is also update when a local function is the best global solution found.

Particles continuously traverse space until the program is terminated, when a final solution has been found (i.e., the global best) and no other solution can be reached.

3. Results

For purpose of calculation, the result were converted from its original type, which is double, to "lowest long bit". The tables below shows result of the experiments performed in "*Standard PSO*" and "*PSO and Crossover*". The figures 3.1 – 3.4 show the flow of best fitness of each run. These results are standardized as maximizing. Each functions optimization was run twenty times. The best fitness value in each run was chosen to calculate the average fitness of a 100 of population of particles during ten run.

As show in table 3.1 and table 3.2, the optimization of *Parabola* function in *"PSO and Crossover"* performed better compared to the Parabola function in "*Standard PSO"*. The *Circle 1* optimization had a better performance in "*Standard PSO"* and in "*PSO and Crossover",* the performance was very poor.

The *Circle 2* optimization in "*Standard PSO"* performed slightly better than the optimization in "*PSO and Crossover".* Meanwhile the Eggbox 2 optimization performed better in "*PSO and Crossover".*

From the pattern of the results during each runs (***Fig 3.1-3.4***), the best possible fitness is the maximum value of 32768.

Function	***Generation***	***Average Fitness***
Parabola	100	9510.4
Circle 1	100	4905.6
Circle 2	100	6756.1
Eggbox 2	100	1456.2

Table 3.1 Result for Standard PSO Experiments

Function	***Generation***	***Average Fitness***
Parabola	100	15276.8
Circle 1	100	625.4
Circle 2	100	6432
Eggbox 2	100	4547.2

Table 3.2 Result for PSO and Crossover Experiments

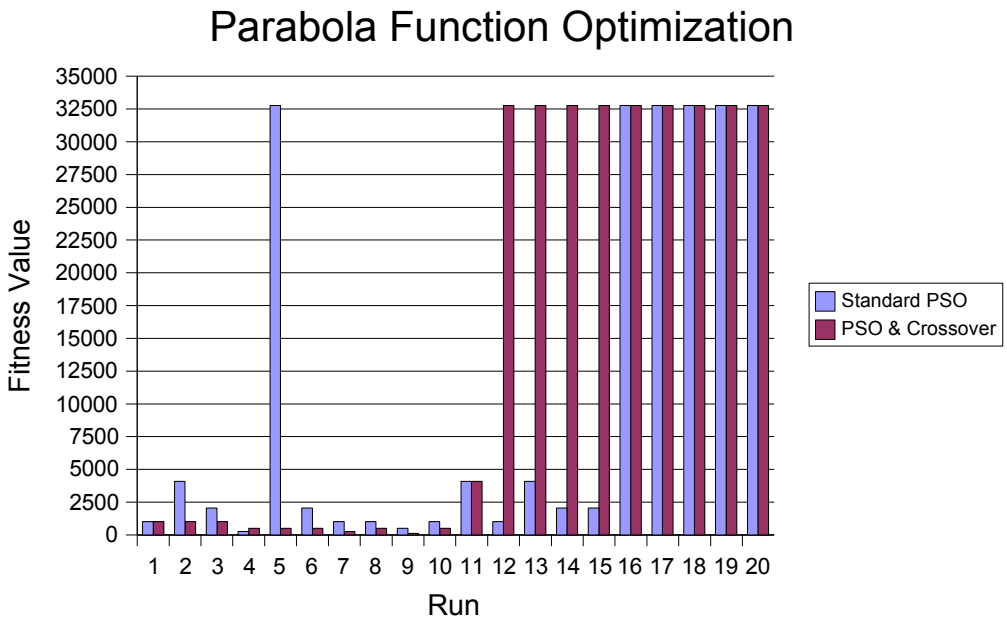

Fig 3.1 Results for Standard PSO & PSO and Crossover Experiments

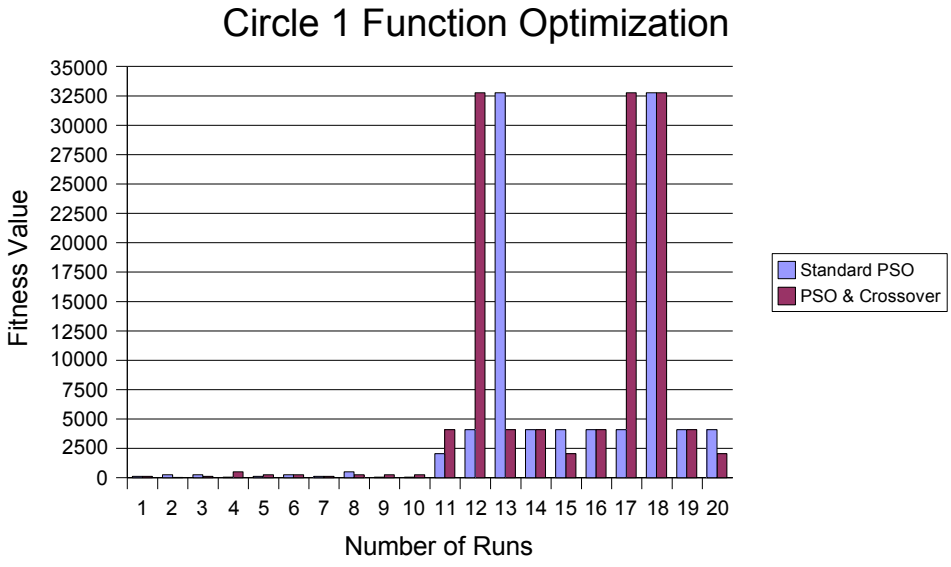

Fig 3.2 Results for Standard PSO & PSO and Crossover Experiments

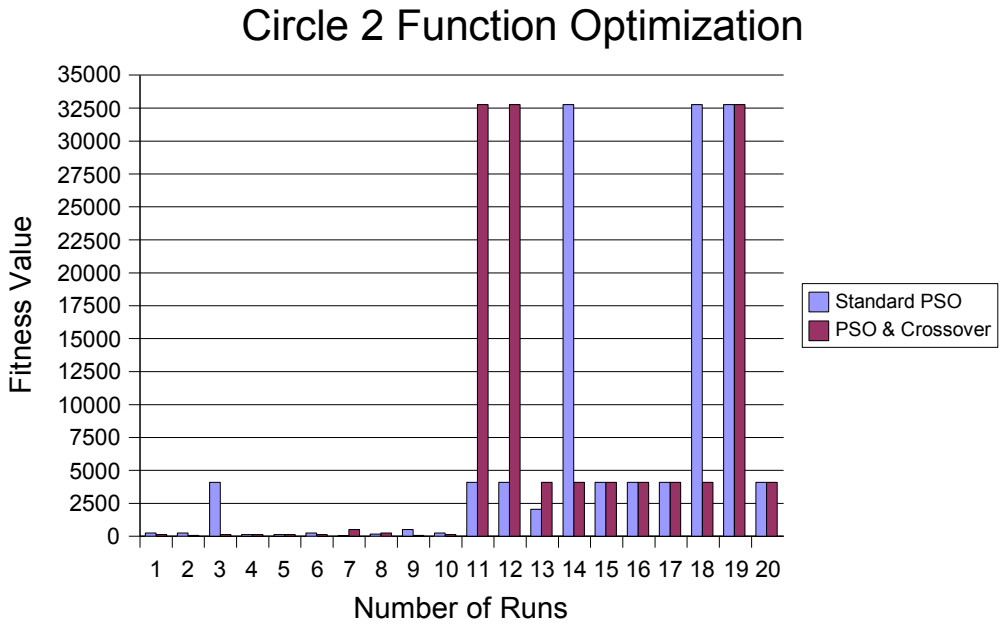

Fig 3.3 Results for Standard PSO & PSO and Crossover Experiments

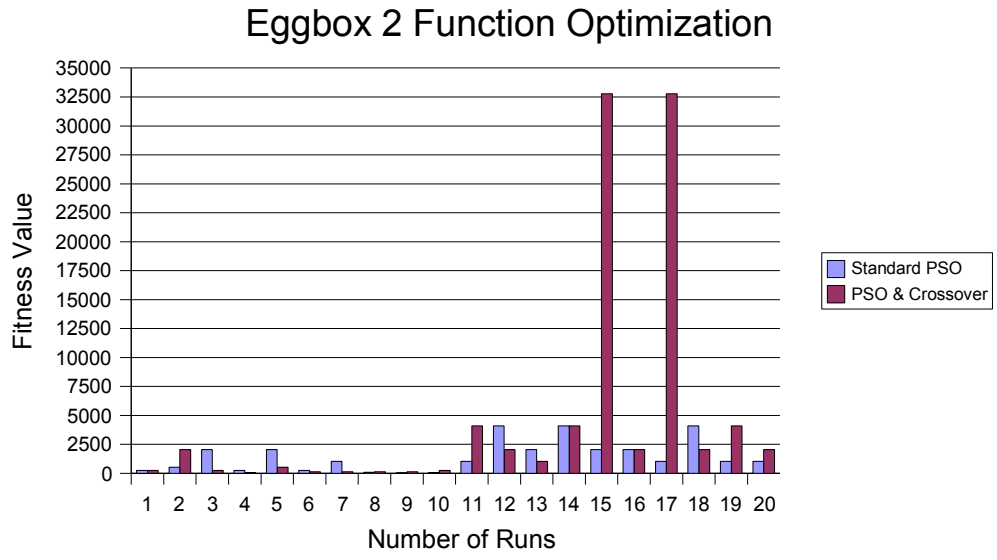

Fig 3.4 Results for Standard PSO & PSO and Crossover Experiments

4. Future Work

The experiment results were based on twenty run and population of 100 particles. Experiments carried out on more population and more runs will give more insight to the average fitness of the particles when in search of global best during the optimization of each function in both *"Standard PSO"* and *"PSO & Crossover*".

Although the swarms benefit from selection, because their "life-time" has been "extended" to search for a better solution, therefore it gains a good fitness value.[7] Since the swarms were judged on their fitness value, the drawback of selection is that some swarm "lifespan" will be "reduced", due to the

fact that they couldn't find a better solution compared to their competitor during the random selection.[7] The selection is based on the survivor of the fittest, which means that those swarm that has “favorable adaptation” will “strive and procreate” whilst the other swarm that has “no favorable adaptation” will have a “reduced life span”.[7]

So performing another PSOpt [6] experiment using another *GA* operator, *Mutation*, is another way of trying to see improvement in *PSO.* Since *Mutation* help to prevent population from stagnating at any local optima, a primitive random search could be performed using Uniform Mutation where the user-specified upper and lower bounds of chosen parents attributes are mutated. Mutation will prevent the swarms attribute from becoming very similar to each other, and the swarm will be able to escape local minima, therefore the lifespan reduction shouldn't be so much.

Comparison between the improvement of “*PSO plus Mutation”*, “*PSO plus Crossover”* and “*PSO plus Crossover plus Mutation”* that all have a larger population, should give more insight to best way to escape local minima and reach a global best in *PSO*.

5. Conclusions

Implementing *Crossover* into *PSO* improve the chances of PSO finding a global best for some given problems, meanwhile in other problems *Standard PSO* works better on its own. The Optimization of *Circle 1* proves *Standard PSO* works better on its own. *Circle 2* optimization also shows that *Standard PSO* performed slightly better than “*PSO and Crossover”*. In “*PSO and Crossover”*, *Parabola* and *Eggbox 2* optimization shows improvement in the fitness of the swarms.

References

1. Angeline P.J. 1998. *Using Selection to Improve Particle Swarm Optimization.* In WCCI-98 Proceedings of the IEEE World Congress on Computational Intelligence pp. 84-89. IEEE Press.
2. Eberhart, R. C., Dobbins, R. W., and Simpson, P. (1996), Computational Intelligence PC Tools, Boston: Academic Press.
3. Eberhart, R. C., and Kennedy, J. (1995). *A New Optimizer Using Particles Swarm Theory, Proc*. Sixth International Symposium on Micro Machine and Human Science (Nagoya, Japan), IEEE Service Center, Piscataway, NJ, 39-43.
4. Kennedy, J., and Eberhart, R. C. (1995). *Particle Swarm Optimization, Proc*. IEEE International Conference on Neural Networks (Perth, Australia), IEEE Service Center, Piscataway, NJ, IV: 1942-1948.
5. Kennedy, J. (1997), *The Particle Swarm: Social Adaptation of Knowledge, Proc*. IEEE International Conference on Evolutionary Computation (Indianapolis, Indiana), IEEE Service Center, Piscataway, NJ, 303-308.
6. Borgelt, Christian, *Particle Swarm Optimization Demonstration*. http://fuzzy.cs.uni-magdeburg.de/%7Eborgelt/psopt.html
7. Tillett, Jason, Rao, T.M., Sahin, Ferat, and Rao, Raghuveer. *Darwinian Particle Swarm Optimization.* University of Rochester.
8. Kennedy, J., Eberhart, R. C., and Shi, Y. (2001). *Swarm Intelligence*, San Francisco. Morgan Kaufmann Publishers.