

Neighborhood Re-Structuring in Particle Swarm Optimization

Arvind S. Mohais¹, Rui Mendes², Christopher Ward¹, and Christian Posthoff¹

¹ The University of the West Indies, St. Augustine, Trinidad, amohais@fsa.uwi.tt

² Universidade do Minho, Braga, Portugal, email: azuki@di.uminho.pt

Abstract. This paper considers the use of randomly generated directed graphs as neighborhoods for particle swarm optimizers (PSO) using fully informed particles (FIPS), together with dynamic changes to the graph during an algorithm run as a diversity-preserving measure. Different graph sizes, constructed with a uniform out-degree were studied with regard to their effect on the performance of the PSO on optimization problems. Comparisons were made with a static random method, as well as with several canonical PSO and FIPS methods. The results indicate that under appropriate parameter settings, the use of random directed graphs with a probabilistic disruptive re-structuring of the graph produces the best results on the test functions considered.

1 Introduction

Particle Swarm Optimization (PSO) [1, 2] is an evolutionary algorithm inspired by social interaction. At each iteration, each particle imitates the behavior of its most successful neighbor. A particle's neighborhood is the set of other particles that it is connected to; it considers their experience when updating its velocity. The graph of inter-connections is called the neighborhood structure. Generally, neighborhood connections are independent of the positions occupied by particles. Two basic structures [1] are the 'global neighborhood', in which each particle is connected to all others and the 'local best' or 'ring', in which particles are connected in a circular manner, with each one being connected to the one on its left and the one on its right. More complex graphs have also been investigated. Mendes [3] studied various fixed graphs on 20 nodes with PSO and conjectured that an average degree of 4 might be suitable for the functions investigated. Using graphs to dictate the interactions between individuals of an evolutionary algorithm has also been used outside of PSO [4].

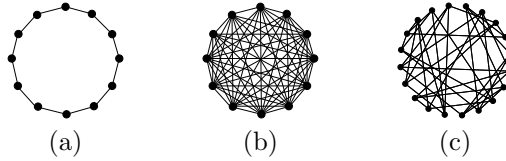
Dynamic neighborhoods, i.e. ones that change the structure of connections as the algorithm progresses have also been studied. Kennedy [5] investigated 'clustering', wherein after each iteration particles form fully connected clusters based on their search-space locations. Suganthan [6] studied the effect of starting with a sparsely connected graph and progressively adding edges during the run, with the aim of gradually shifting the focus from exploration to exploitation. Mohais et al [7] performed a proof-of-concept study involving the random migration of edges in a randomly-generated graph, meaning that some edges were disconnected from one of their endpoints and randomly re-attached elsewhere.

This paper further studies random and dynamic neighborhoods. Different size populations, of uniform out-degree 5 were tested on six functions. Two types of probabilistic dynamism were considered: random edge migrations and complete graph re-structuring. These methods were also compared to some predefined static neighborhood structures, L-Best-1 (the ring), L-Best-2, and the von Neumann topology.

2 Particle Swarm Optimization

A Particle Swarm Optimizer is a set, or population, of n particles that evolves over time. At time t , the population is $Pop(t) = \{P_1(t), P_2(t), \dots, P_n(t)\}$ where the i^{th} particle is defined by $P_i(t) = (x_i(t), p_i(t), v_i(t), e_i(t))$. The current position in the d -dimensional search space is $x_i(t) \in \mathbb{R}^d$. $p_i(t) \in \mathbb{R}^d$ is the position visited in the past that had the best function evaluation. $v_i(t) \in \mathbb{R}^d$ is called the ‘velocity’, it is the speed and direction with which the particle is moving. $e_i(t)$ is the evaluation of $p_i(t)$ under the function being optimized, i.e. $e_i(t) = f(p_i(t))$.

Particles are connected to others in the population via a predefined topology. Common topologies are the ring and global topologies which are illustrated below ((a) and (b)). However there is no reason that an asymmetrical, or even randomly generated topology cannot be used (see (c)).



At each iteration, a new population is produced by allowing each particle to move stochastically toward its previous best position and at the same time toward the best of the previous best positions of all other particles to which it is connected. The following is an outline of a generic PSO.

1. Set the iteration counter, $t = 0$.
2. Initialize each $x_i(0)$ and $v_i(0)$ randomly. Set $p_i(0) = x_i(0)$.
3. Evaluate each particle and set $e_i(0) = f(p_i(0))$.
4. Let $t = t + 1$ and generate a new population, $P(t)$ by moving each particle i in $P(t - 1)$ to a new position in the search space according to:
 - (i) $v_i(t) = velocity_update(t - 1)$.
 - (ii) $x_i(t) = x_i(t - 1) + v_i(t)$.
 - (iii) Evaluate the new position, $e = f(x_i(t))$. If it is better than the previous best, update the particle’s previous best position. i.e if $e < e_i(t - 1)$ then let $p_i(t) = x_i(t)$ and $e_i(t) = e$, else let $p_i(t) = p_i(t - 1)$ and $e_i(t) = e_i(t - 1)$.

The original PSO used the following velocity update equation

$$velocity_update(t - 1) = v_i(t - 1) + r_1 c_1 (p_i(t - 1) - x_i(t - 1)) + r_2 c_2 (p_g(t - 1) - x_i(t - 1))$$

where c_1 and c_2 are constants known as the ‘individual’ and ‘social’ constants respectively. They represent the weights accorded to the influence of the particle’s personal memory, and the memory of it’s neighborhood respectively. $r_1, r_2 \sim U(0, 1)$. New values for r_1 and r_2 are selected for each dimension of the updated velocity vector as it is being computed. $p_g(t-1)$ is the previous best position of the particle in i ’s neighborhood that has the best previous best evaluation of all particles in that neighborhood at time $t-1$. In other words, $g = \arg\{\min\{e_j(t-1) | j \in N(i)\}\}$, where $N(i)$ is the neighborhood of particle i . Two variations on the velocity update equation are described below.

Constriction Factor PSO (canonical) Clerc and Kennedy [8] introduced the use of a ‘constriction factor’ χ , into the velocity update equation.

$$velocity_update(t-1) = \chi[v_i(t-1) + r_1 c_1 (p_i(t-1) - x_i(t-1)) + r_2 c_2 (p_g(t-1) - x_i(t-1))]$$

A common configuration involves choosing $c_1 = c_2 = 2.05$ and $\chi = 0.729$.

Fully Informed PSO (FIPS) An approach that involves utilizing information from all members of particle i ’s neighborhood, $N(i)$, was proposed by Mendes et al [9]. Each member of $N(i)$ contributes to the new velocity. Mendes’ formulation allows for weighted (possibly equal) contributions from each neighbor. With $\phi_{max} = c_1 + c_2$, the velocity update equation becomes,

$$velocity_update(t-1) = \chi \left[v_i(t-1) + \frac{\sum_{k \in N(i)} U(0, \phi_{max})(p_k - x_t)}{|N(i)|} \right]$$

3 Random and Dynamic Neighborhoods

In this paper, directed graphs were used to represent neighborhoods. An edge from u to v was taken to mean that u will consider v as a neighbor, but not vice versa. Two parameters were used in the generation of a random neighborhood. The first was the size n of the neighborhood, and the other was the out-degree of each node (the number of outgoing edges). Based on preliminary experimental trials, a uniform out-degree of 5 was used. For each node, 5 different, randomly selected neighbors, were added to the node’s neighborhood.

Two methods of modifying the structure of a neighborhood were considered. The first is called ‘random edge migration’ [7]. It involves randomly selecting a node with a neighborhood of size greater than 1, detaching one of its neighbors from it, and re-attaching that neighbor to some other randomly selected node that does not have a full neighborhood.

The second method is called a ‘neighborhood re-structuring’. This is an entirely new method that involves a more abrupt type of dynamism in the neighborhood structure. Instead of there being small and gradual changes, this approach keeps the neighborhood structure fixed for an amount of time and then

completely re-initializes it; essentially imposing a new random neighborhood structure on the population. Only the parameters of the neighborhood, i.e. size and out-degree, are kept fixed, the connections are completely changed.

4 Experimental Investigations

4.1 The Test Problems

The neighborhood structures and dynamic modification schemes described above along with standard schemes were tested on six optimization functions commonly used by researchers of PSO systems. The functions are given below. The Schaffer F6 function is 2 dimensional, all the others were in 30 dimensions except the Griewank function which was used in 10 dimensions since in practice this is harder to optimize than the 30-dimensional one.

Sphere	$\sum_{i=1}^n x_i^2$
Rosenbrock	$\sum_{i=1}^{n-1} 100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2$
Ackley	$20 + e - 20 e^{-0.2\sqrt{\frac{\sum_{i=1}^n x_i^2}{n}}} - e^{\frac{\sum_{i=1}^n \cos 2\pi x_i}{n}}$
Rastrigin	$\sum_{i=1}^n x_i^2 - 10 \cos 2\pi x_i + 10$
Griewank	$1 + \frac{\sum_{i=1}^n (x_i - 100)^2}{4000} - \prod_{i=1}^n \cos \frac{x_i - 100}{\sqrt{i}}$
Schaffer F6	$0.5 + \frac{\sin(\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1 + 0.001(x_1^2 + x_2^2))^2}$

4.2 General Experimental Protocol

The experiments involved configuring a particular type of PSO, and testing it on each of the six optimization problems. Each configuration was tested 50 times on each function. At the end of each run, the global best function evaluation was recorded. Each run was allowed to run for a maximum of 20000 function evaluations. This is equivalent to using a population size of 20 for 1000 iterations. However, for different size populations, fewer or more iterations would result.

4.3 Experiment 1: Comparison of Random Dynamic Schemes

A first experiment compared the performance of re-structuring, migration and, static random neighborhoods. For all experiments an initial random neighborhood was generated; sizes ranged from 10 to 100 in steps of 5 and regardless of the size, each node was randomly assigned 5 neighbors. An out-degree of 5 was chosen because initial probing suggested it to be a good choice.

For the migration method, a probability parameter $p_m \in \{0.1, 0.2, \dots, 1.0\}$ controlled how often an edge migration would occur. After each iteration, a

decision was made whether or not to perform a random edge migration, based on the p_m value. Similarly, for re-structuring, after each iteration a decision was made whether or not to perform the operation based on a probability parameter $p_r \in \{0.01, 0.02, \dots, 0.1\}$.

4.4 Results of Experiment 1

A first observation is that a distinctive shape is obtained when population size is plotted against mean best performance, regardless of whether the static method or a dynamic one was used. Figure 1 illustrates this, each plot contains overlays of the population size vs mean best performance plots of the static method, and all probabilities of the migration and re-structuring methods for a given function.

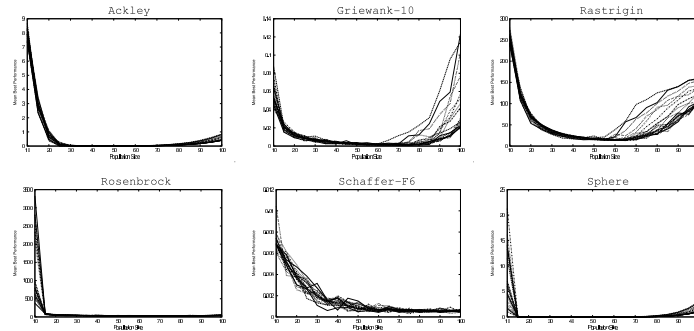


Fig. 1. Overlays of population size versus mean best performance plots for PSOs using random neighborhoods either statically or dynamically.

Table 1 shows the top three configurations in each method, the migration configuration using a probability p_m is represented by ‘mig- p_m ’ and the re-structuring configuration using a probability p_r is represented by ‘rst- p_r ’. The results are sorted by mean best performance.

4.5 Experiment 2: Comparison with other methods

A second experiment was performed to compare the best of these methods to a state-of-the-art variant of the PSO algorithm, as well as with some standard PSO configurations. Each PSO variant described below was tested using population sizes of 10, . . . , 100 in steps of 5.

Popsiz Variant			Performance			Popsiz Variant			Performance		
Ackley						Griewank-10					
25	rst-0.10	1.32E-7	±	3.23E-8	70	rst-0.04	4.96E-4	±	6.69E-4		
25	rst-0.09	1.65E-7	±	4.69E-8	75	mig-0.8	5.63E-4	±	5.28E-4		
30	mig-0.5	1.06E-6	±	1.53E-7	75	rst-0.01	6.30E-4	±	5.93E-4		
30	mig-0.7	1.29E-6	±	1.87E-7	75	mig-0.5	6.82E-4	±	5.97E-4		
30	mig-0.1	1.48E-6	±	3.81E-7	70	mig-0.3	7.22E-4	±	6.81E-4		
30	rst-0.05	1.52E-6	±	2.24E-7	55	rst-0.02	8.87E-4	±	9.25E-4		
40	static	7.92E-5	±	7.68E-6	85	static	1.54E-3	±	6.84E-4		
45	static	3.08E-4	±	2.67E-5	60	static	1.76E-3	±	1.21E-3		
50	static	1.02E-3	±	7.36E-5	65	static	2.12E-3	±	1.34E-3		
Rastrigin						Rosenbrock					
65	mig-0.1	12.860	±	1.234	45	rst-0.06	26.560	±	0.192		
65	mig-0.5	13.078	±	1.053	50	mig-1.0	26.760	±	0.084		
60	mig-0.9	13.092	±	1.076	50	rst-0.09	26.933	±	0.130		
60	rst-0.01	13.184	±	1.141	55	rst-0.08	27.257	±	0.118		
55	rst-0.07	13.184	±	1.086	45	mig-0.3	27.628	±	2.358		
60	rst-0.04	13.325	±	1.157	65	mig-1.0	27.692	±	0.067		
65	static	13.580	±	1.226	75	static	28.323	±	0.084		
60	static	13.603	±	1.076	85	static	29.609	±	0.162		
55	static	14.657	±	1.002	60	static	30.356	±	4.274		
Schaffer-F6						Sphere					
75	rst-0.05	2.99E-4	±	9.22E-5	20	rst-0.10	8.55E-15	±	6.77E-15		
95	mig-0.7	3.32E-4	±	1.15E-4	20	rst-0.09	1.96E-14	±	1.89E-14		
75	rst-0.08	4.01E-4	±	1.02E-4	20	rst-0.03	7.35E-14	±	5.52E-14		
80	rst-0.07	4.03E-4	±	9.98E-5	20	mig-1.0	3.49E-13	±	3.40E-13		
90	mig-0.3	4.22E-4	±	1.64E-4	25	mig-0.6	3.85E-13	±	2.20E-13		
90	mig-1.0	4.29E-4	±	1.21E-4	25	mig-0.9	4.09E-13	±	1.97E-13		
100	static	4.32E-4	±	1.13E-4	25	static	9.17E-12	±	1.39E-11		
95	static	5.62E-4	±	2.25E-4	30	static	5.62E-11	±	2.01E-11		
70	static	5.65E-4	±	1.92E-4	20	static	5.19E-10	±	1.04E-9		

Table 1. Top 3 configurations, in terms of mean best performance, for each random neighborhood method (migration, re-structuring and static), given for each function

It has been found that a PSO using FIPS with the von Neumann (a toroidal grid) neighborhood is usually a good choice for the optimization functions being considered. Thus, this was one of the PSO variants used for comparison in this experiment. Given a population size of n , the von Neumann neighborhood was configured into r rows and c columns, where r is the smallest integer less than or equal to \sqrt{n} that evenly divides n and $c = n/r$. L-Best-1 and L-Best-2 were also tested with FIPS. The L-best- k topology consists of n nodes arranged in a ring, in which node i is connected to each node in $\{(i + j) \bmod n : j = \pm 1, \pm 2, \dots, \pm k\}$. Some other standard PSO configurations were used; these were the canonical constriction factor PSO using the L-best-1, L-best-2 and von Neumann topologies. Finally the global neighborhood (gbest) was tested with the canonical PSO.

4.6 Results of Experiment 2

Table 2 summarizes the top performing configurations of the methods used for comparison. Only the best three methods out of the seven tested are shown. A prefix of ‘fips’ means FIPS was used, otherwise a canonical PSO was used; ‘lb1’, ‘lb2’ and ‘von’ stand for L-Best-1, L-Best-2 and von Neumann respectively.

Popsiz		Variant		Performance		Popsiz		Variant		Performance	
Ackley						Griewank-10					
25	fips_lb2	2.83E-5	± 3.80E-6	40	fips_lb2	4.82E-3	± 2.53E-3	40	fips_von	5.40E-3	± 3.15E-3
25	fips_von	2.92E-5	± 5.19E-6	40	fips_von	5.40E-3	± 3.15E-3	30	fips_lb1	8.04E-3	± 2.90E-3
25	fips_lb1	5.64E-5	± 1.33E-5								
Rastrigin						Rosenbrock					
30	fips_von	20.270	± 1.843	40	fips_lb2	28.566	± 2.225	55	fips_von	33.119	± 3.811
30	fips_lb1	20.566	± 1.530	40	fips_lb1	35.293	± 7.976				
30	fips_lb2	20.835	± 2.070								
Schaffer-F6						Sphere					
35	von	3.81E-5	± 4.61E-5	15	fips_lb2	1.94E-11	± 2.94E-11	20	fips_von	6.31E-11	± 4.17E-11
40	fips_lb1	3.81E-4	± 1.73E-4	15	fips_lb1	2.02E-10	± 2.52E-10				
100	fips_lb2	4.86E-4	± 1.78E-4								

Table 2. Top configurations in terms of mean best performance, of the schemes tested in experiment 2, given for each function. Only the best 3 are shown.

5 Analysis of Results

5.1 Methodology

When conducting experiments, it is necessary to follow a methodology that will validate the results. We often want to know if there is a significant difference between several approaches. The Wilcoxon rank-sum test is a non-parametric alternative to the two-sample t -test which is based solely on the order in which the observations from the two samples fall. It is valid for data from any distribution, Normal or not, and is much less sensitive to outliers than the two-sample t -test. When the assumptions of the two-sample t -test hold, the Wilcoxon test is somewhat less likely to detect a location shift than is the two-sample t -test. However, the losses in this regard are usually quite small [10].

We used Wilcoxon tests for two-sample comparisons and Kruskal-Wallis comparisons for 3 or more independent samples. The Kruskal-Wallis test is a non-parametric alternative to the one-way independent-samples ANOVA. Our results are presented using a symbolic encoding of the p -values, this improves readability and allows us to visually interpret the result of a test. The p -values were coded as follows: **3** if $p < 0.01$; **2** if $0.01 \leq p < 0.05$; **1** if $0.05 \leq p < 0.1$ and **-** if $p \geq 0.1$.

The standard cut-off for considering a p -value for a statistically significant difference is $p < 0.05$. For a single comparison, this indicates that the observed

differences would occur by chance only 5% of the time. The probability of observing a sizable difference for one of the comparisons increases with the number of comparisons performed. For 20 independent comparisons, the probability of the observed differences occurring by chance increases to 64%. This is why it is important to correct the p -value of each test when performing multiple comparisons.

There are several p -value corrections that may be used. One of the simplest and common ones is the Bonferroni correction. It is very simple to apply: one simply multiplies each p -value by the number of tests performed. As a consequence, when performing 20 tests at a time, the highest accepted individual p -value is 0.0025. The main concern of this correction is to control the number of false positives. However, it does so at the expense of many false negatives.

The Bonferroni step-down (Holm) correction is very similar to the Bonferroni, but a little less stringent [11]. The p -values of each test are ranked from the smallest to the largest. The smallest p -value is multiplied by the number of tests, t , the next by $t - 1$, the one after that by $t - 2$ and so on. Because it is a little less corrective as the p -value increases, this correction is less conservative. However, the family-wise error rate is very similar to the Bonferroni correction. We used the Holm correction in our analysis.

5.2 Discussion of the Results

The dynamic methods performed well, but different population sizes and dynamism probabilities are required in order to obtain such performance for different functions. The plots for Ackley, Griewank-10, Rastrigin and Sphere in figure 1 show that there is a basin-shaped region in which to find the best population size for these functions. Performance is best near the bottom of this basin; it is poor for small as well as for very large population sizes. The plots for all three random neighborhood methods follow the same general shape regardless of the probabilities used. Unfortunately, the widths and locations of these basins for the various functions are different. The Rosenbrock and Schaffer-F6 functions differ from this trend in that larger population sizes seem to give better performance.

Tables 1 and 2 suggest that the best configuration of re-structuring always generates the best performance in all functions. We performed pairwise two-tailed Wilcoxon tests using the Holm correction (table 3). This in conjunction with tables 1 and 2 shows that the best performance of re-structuring was indeed superior to the other methods on the Ackley, Rosenbrock and Sphere functions. It was better than the non-random methods on the Rastrigin function. On Griewank-10 it was better than the migration and static random methods, but no significant difference was found in comparison to the non-random methods. No significant difference was found with any of the other methods on Schaffer-F6.

Overall, the results suggest that using directed random graphs and dynamically changing the graph by probabilistic re-structuring improves the robustness of the algorithm in comparison to using non-random neighborhoods. Statistical evidence for this was obtained in 4 out of the 6 functions. We could not find a

statistically significant difference for the other two. The results did not provide any indication of how to configure the random graphs or the rate of dynamism.

We decided to test for the importance of the effect of both the probability and population size parameters on the restructuring method. Table 4 shows the results of the two-sided Kruskal-Wallis tests of the effect of the probability of the restructuring method; the values were corrected using the Holm method. In most cases the difference is significant. The exceptions where the Schaffer function, Rosenbrock for all the population sizes below 50 and Griewank below 30. A similar test was conducted to account for the effect of population sizes, grouped by probability. As all p -values were below 0.01, we decided not to present the results in a table.

	Wilcoxon Signed Rank Test of Re-Structuring Versus				
	Migration	Static	von Neumann	L-Best-1	L-Best-2
Ackley	3	3	3	3	3
Griewank-10	3	3	-	-	-
Rastrigin	-	-	3	3	3
Rosenbrock	3	3	3	3	3
Schaffer-F6	-	-	-	-	-
Sphere	3	3	3	3	3

Table 3. Wilcoxon signed rank tests. The table compares re-structuring with other methods. von Neumann, L-Best-1 and L-Best-2 were used with FIPS.

	Kruskal-Wallis Rank Sum Tests by Pop. Sizes																			
	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	
Ackley	-	1	3	-	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3
Griewank-10	-	-	-	-	-	3	3	3	-	3	3	3	3	3	3	3	3	3	3	3
Rastrigin	-	1	3	2	3	3	3	3	-	-	3	3	3	3	3	3	3	3	3	3
Rosenbrock	-	-	-	-	-	-	-	-	-	3	2	3	3	3	3	3	3	3	3	3
Schaffer-F6	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Sphere	1	3	3	1	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Table 4. Kruskal-Wallis tests for the re-structuring method. Groupings consist of all probabilities for a given population size.

6 Conclusions

This paper looked at the effect of using random directed graphs with a uniform outdegree of five, in PSOs. Once generated at the beginning of the algorithm, the graphs were then either kept fixed throughout the run, or were modified during

the run. Two forms of modification were considered; the first was a gradual type in which, probabilistically at each iteration, a randomly selected edge was moved in the graph (migration). The second was one in which the entire graph was re-initialized probabilistically (re-structuring) at each iteration. These methods were tested on a PSO using the fully informed neighborhood technique. Tests were performed using a benchmark of six optimization functions. The results were validated against commonly used and state of the art approaches in PSO.

The results indicate that probabilistic re-structuring is a good strategy for social topologies. On each function, our approach is never worse than any other, and in most cases it is the best. In most instances, statistical evidence was found to prove a clear superiority over the non-random methods. In two cases no such evidence was found; high p -values indicated that there was no statistical difference between re-structuring and the non-random methods. It was found that the interaction of population size and re-structuring probability affects the performance of the algorithm in most cases.

Previous studies show that the correct choice of a specific fixed graph for use in PSO gives good results, but these graphs are difficult to code and to generalize. The fact that the random choice of social topologies works well indicates that there is no need for such fixed topologies. Further research is needed to improve these methods, especially to create rules about population sizes and dynamism probabilities. The reason for the performance of random dynamic topologies is an important outstanding question.

References

1. Kennedy, J., Eberhart, R.: Particle swarm optimization. In: Proc. of IEEE International Conference on Neural Networks, IEEE Press (1995) 1942–1948
2. Kennedy, J., Eberhart, R., Shi, Y.: Swarm Intelligence. Morgan Kaufmann (2001)
3. Mendes, R.: Population Topologies and Their Influence in Particle Swarm Performance. PhD thesis, Universidade do Minho, Braga, Portugal (2004)
4. Ashlock, D., Smucker, M., Walker, J.: Graph based genetic algorithms. In: Proc. of the IEEE Congress on Evolutionary Computation, IEEE Press (1999) 1362–1368
5. Kennedy, J.: Stereotyping: Improving particle swarm performance with cluster analysis. In: Proc. of the IEEE Congress on Evolutionary Computation. (2000) 1507–1512
6. Suganthan, P.N.: Particle swarm optimiser with neighbourhood operator. In: Proc. of the IEEE Congress on Evolutionary Computation, IEEE Press (1999) 1958–1962
7. Mohais, A., Ward, C., Posthoff, C.: Randomized directed neighborhoods with edge migration in particle swarm optimization. In: Proc. of the 2004 IEEE Congress on Evolutionary Computation, IEEE Press (2004) 548–555
8. Clerc, M., Kennedy, J.: The particle swarm - explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation* **6** (2002) 58–73
9. Mendes, R., Kennedy, J., Neves, J.: The fully informed particle swarm: Simple, maybe better. *IEEE Transactions on Evolutionary Computation* **8** (2004) 204–210
10. Gouri K. Bhattacharyya, R.A.J.: Statistical Concepts and Methods. (May 1977)
11. Holm, S.: A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics* **6** (1979) 65–70