Watch Thy Neighbor Or How The Swarm Can Learn From Its Environment

Rui Mendes[1], James Kennedy[2], and José Neves[1]

[1] Departamento de Informática
Universidade do Minho
Braga, Portugal
{azuki, jneves}@di.uminho.pt
[2] Bureau of Labor Statistics
Washington DC
Kennedy_Jim@bls.gov

Abstract. Particle Swarm Optimization is a novel algorithm where a population of candidate problem solution vectors evolves "social" norms by being influenced by their topological neighbors. Until now, an individual was influenced by its best performance acquired in the past and the best experience observed in its neighborhood. In this paper, we introduce new ways an individual can be influenced by its neighbors.

1 Introduction

Recently, the importance of population topologies in particle swarm performance has been demonstrated [3] [2]. Experimentation with various topologies, along with indepth analysis of the trajectories of individual members of a particle swarm population [1] [4] [6], has resulted in improvements in the performance of the algorithm. However, we feel that there is still more ground to cover, especially when it comes to the way an individual is influenced by his neighborhood.

The canonical particle swarm can be described as a population of vectors whose trajectories oscillate around a region defined by each individual's previous best success and the success of the single best particle in its neighborhood. In this paper, our aim is to extend that notion, incorporating information of the best positions found by all the particles in the neighborhood. The rationale is that when looking for success, an individual is not only influenced by the best performer in his entourage, but by some aspects of all individuals observed.

Drawing on the social-psychological analogy, it seems reasonable to think that the strength of an individual's influence depends on his success, i.e., a successful individual influences our decisions more than a less successful one. Therefore, we present two models, both combining information about the sucess of the particles in the neighborhood. One of the models equally regards each particle and the other weighs each contribution according to its importance.

Section 2 describes the basics of particle swarm and presents our approach, section 3 presents the neighborhoods used, section 4 presents the functions to minimize, used to compare the algorithms, the results of the experiments and discusses them, finally, the last section presents the conclusions.

2 The algorithms

In the traditional particle swarm, each particle explores around a region defined by its previous best success and the success of the best particle in its neighborhood. The canonical equation, using type 1″ constriction as proposed by Clerc in [1], describing the velocity and position of particle i is given as follows:

$$v_{t+1} = \chi(v_t + \varphi(P_m - X_t)) \tag{1}$$

$$X_{t+1} = X_t + v_{t+1} \tag{2}$$

for each coordinate, with v_t being the current velocity along that coordinate, X_t being the particle's position, $\varphi_{max} = 4.1$ and $\chi = 0.729844$. P_m represents the point in search space a particle will converge to. Traditionally, P_m is calculated using the best position attained by the particle i (P_i) and the best position found by its neighborhood (P_b). Hence, P_m and φ are calculated as follows:

$$\varphi_1 = \mathrm{U}[0, \frac{\varphi_{max}}{2}] \tag{3}$$

$$\varphi_2 = \mathrm{U}[0, \frac{\varphi_{max}}{2}] \tag{4}$$

$$\varphi = \varphi_1 + \varphi_2 \tag{5}$$

$$P_m = \frac{\varphi_1 P_i + \varphi_2 P_b}{\varphi} \tag{6}$$

In this paper, we suggest that the individual should gather information about the whole neighborhood. For that, let us define $\mathcal{N}$ as the set of neighbors of i, P_k as the best position found by individual k and f_k as the fitness (or quality) of that position. Without loss of generality, and to keep the equations simple (without need of normalization) we will only consider minimization tasks and problems with only positive fitness values. As such, a lower f_k represents a better solution. We consider two approaches, one where each solution contributes equally and another where the quality of the solution is also considered. For the first approach, we have:

$$\varphi_k = \mathrm{U}\left[0, \frac{\varphi_{max}}{|\mathcal{N}|}\right] \ \forall k \in \mathcal{N} \tag{7}$$

$$\varphi = \sum_{k \in \mathcal{N}} \varphi_k \tag{8}$$

$$P_m = \frac{\sum_{k \in \mathcal{N}} \varphi_k P_k}{\varphi} \tag{9}$$

In the approach where the quality of the solutions is considered, P_m is calculated as:

$$P_m = \frac{\sum_{k \in \mathcal{N}} \frac{\varphi_k P_k}{f_i}}{\sum_{k \in \mathcal{N}} \frac{\varphi_k}{f_i}} \tag{10}$$

In the rest of the paper, PSO will refer to the traditional method, PSON to the first approach proposed (where each solution is considered equally) and PSOWN will refer to the second approach.

3 Population Topologies

As was shown in [3], the importance of population topologies is paramount. In this study, we present the topologies we have designed. Several sociometries were studied (see figure 1), all with population size of 20. These were

gbest which treats the entire population as the individual's neighborhood.

lbest where adjacent members of the population array comprise the neighborhood.

pyramid a three-dimensional wire-frame triangle.

von Neumann a lattice whose extremities connect as a torus.

four clusters four clusters, completely interconnected, connected among themselves by a few shortcuts.

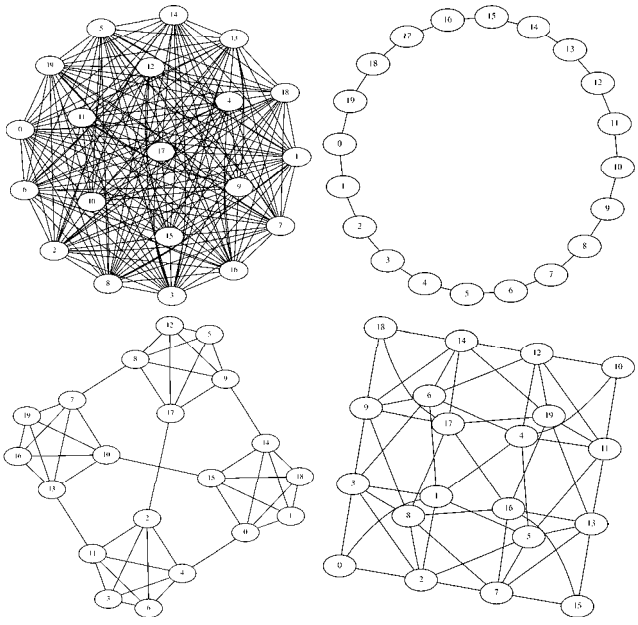

Fig. 1. Some of the topologies used: gbest, lbest, four clusters and pyramid, respectively.

To study the importance of including the past experiences of an individual, these sociometries were implemented with and without self included, except for *four clusters.*

4 Experimentation

We use two measures of performance. The first is the best function result attained after a certain number of iterations. In our case, we report the best result found after 1000 iterations.

It is possible that an algorithm will rapidly reach a relatively good result while becoming trapped on a local optimum. Thus, a second measure used is the number of iterations needed for the algorithm to meet a criterion. The criteria are given in Table 1. The algorithms were run for 10000 iterations or until the criterion was met. If it was not met by that time, the measure was considered infinite, that is, it was reported as if the criterion would never be met. Thus, medians are reported rather than means for iteration results.

A third dependent measure was derived from the second. That is a simple binary variable describing whether the criterion was attained within 10000 iterations or not.

For each configuration of algorithm and topology, 40 runs were used on each function to gather the data necessary to compare the different approaches. The results presented are the result of those tests.

4.1 Functions

Five standard test functions were employed in the present research. These were the Sphere function, Rastrigin, Griewank, Rosenbrock and Schaffer's f6 [5]. Table 1 shows the parameters for each function used. Two instances of the Griewank function were used because both are quite chalenging. In fact, Griewank in 10 dimensions presents a serious problem for the canonical PSO, and was one of the motivations of our quest for variants that could overcome it.

Function	Dimensions	Initial Range	Criterion
Sphere	30	±100	0.01
Rastrigin	30	±5.12	100
Griewank10	10	±600	0.05
Griewank30	30	±600	0.05
Rosenbrock	30	±30	100
Schaffer f6	2	±100	0.00001

Table 1. Parameters and Criteria for the test functions

Sphere $f_1(x)$ is a very simple, unimodal function. The minimum is $f_1(0) = 0$.

$$f_1(x) = \sum_{i=1}^{n} x_i^2; \qquad x_i \in [-100, 100]$$

Rastrigin $f_2(x)$ is a multimodal version of the Sphere, characterized by deep local minima arranged as sinusoidal bumps. The global minima is $f_2(0) = 0$.

$$f_2(x) = \sum_{i=1}^{n} x_i^2 - 10\cos(2\pi x_i) + 10; \qquad x_i \in [-5.12, 5.12]$$

Griewank $f_3(x)$ has many local optima, whose importance diminishes when the number of dimensions rises. It is very difficult to find the true minimum at $f_3(0) = 0$.

$$f_3(x) = \sum_{i=1}^{n} \frac{(x_i - 100)^2}{4000} - \prod_{i=1}^{10} \cos\left(\frac{x_i - 100}{\sqrt{i}}\right) + 1; \qquad x_i \in [-600, 600]$$

Rosenbrock $f_4(x)$ is extremely steep when the optimum is being approached from afar and banana shaped close to the optimum.

$$f_4(x) = \sum_{i=1}^{n-1} 100\,(x_{i+1} - x_i^2)^2 + (x_i - 1)^2; \qquad x_i \in [-30, 30]$$

Schaffer f6 $f_5(x)$ is a very difficult function, especially devised to trick optimization algorithms with its many local optima arranged in concentric circles around the global optimum that is itself located in a narrow basin, making it very hard to reach the global optima at $f(0) = 0$.

$$f_5(x) = 0.5 + \frac{\sin(\sqrt{x_1^2 + x_2^2})^2 - 0.5}{(1 + 0.001\,(x_1^2 + x_2^2))^2} \qquad x_i \in [-100, 100]$$

4.2 Methodology

We believe that it is important to find a method that will be able to achieve good performance on all the functions tested. The No Free Lunch theorem [7] [8] asserts that no algorithm can perform better than any other, averaged across all possible functions. We do think it desirable though to find a problem-solver that performs well on a variety of functions that have been identified as hard problems. To achieve this goal, it is necessary to have a method capable of comparing values of different magnitude.

We standardized the results of the tests over each of the functions. A standardized result X_s is computed according to formula (11), where μ is the average of the values and σ is the standard deviation. A value below zero is below average and a positive one is above average. Using standardization, it is possible to compare algorithms by the average of the normalized results obtained over all the functions.

$$X_s = \frac{X - \mu}{\sigma} \tag{11}$$

4.3 Results

To clarify the visualization of the results, we show the rankings of the different configurations instead of the standardized values. Table 2 shows the results, where the fields are proportion of runs that reached the criteria and respective rank (PropRank), ranking of median number of iterations to reach the criteria (CritRank) and ranking of average performance at 1000 iterations (PerfRank). Note that the neighborhoods whose name starts with *s* correspond to the version with the self included, i.e. where the particle belongs to its own neighborhood.

Method	Neighborhood	Proportion	PropRank	CritRank	PerfRank
PSOWN	lbest	100.00%	1	19	22
PSON	vneumm	98.75%	2	4	1
PSON	slbest	98.75%	2	11	14
PSOWN	slbest	98.33%	4	5	6
PSON	lbest	96.67%	5	22	23
PSOWN	vneumm	96.25%	6	3	2
PSO	vneumm	92.50%	7	10	10
PSO	four_clusters	92.08%	8	9	15
PSON	pyramid	91.67%	9	6	5
PSOWN	four_clusters	91.67%	9	2	9
PSON	four_clusters	91.25%	11	7	7
PSO	slbest	91.25%	11	13	20
PSO	lbest	90.83%	13	12	12
PSOWN	pyramid	90.00%	14	1	8
PSO	svneumm	87.50%	15	18	16
PSO	pyramid	87.08%	16	17	13
PSOWN	svneumm	86.67%	17	8	3
PSO	spyramid	85.42%	18	16	17
PSON	svneumm	81.25%	19	20	4
PSOWN	spyramid	78.33%	20	21	11
PSO	gbest	75.42%	21	15	19
PSO	sgbest	75.42%	21	14	21
PSON	spyramid	70.42%	23	23	18
PSOWN	gbest	22.08%	24	25	25
PSOWN	sgbest	18.33%	25	26	26
PSON	gbest	16.67%	26	24	27
PSON	sgbest	13.75%	27	27	24

Table 2. Results of the tests

The configuration that was able to reach the criteria every time was PSOWN with the *lbest* neighborhood. However, it was 22th in performance. In other words, this variant consistently found the global optimum, but took a long time getting there. If performance is considered more important, that is, the best function result at 1,000 iterations, then the best configuration is PSON with

the *von Neumann* neighborhood. This variant also has a very high proportion of sucess (it is the second in ranking with a proportion of 98.75%) and is one of the fastest algorithms, ranking fourth in the median number of iterations to reach the criteria.

The best performing neighborhoods over all the algorithms are *lbest* (in terms of proportion of success) and *von Neumann*, in terms of both proportion and performance. The latter is an overall performer, and should be considered as a good alternative to the sociometries used so far. *Gbest* is a bad performer on PSON and PSOWN. The reason for this is that if each individual is influenced by a considerable part of the population, then the information derived from the population is the same, the center of gravity of the population, which inhibits exploration because of lack of diversity. In a sociometry where the neighborhood of each individual is small, each of the individuals will have a different vision of the search space, and therefore will have a better performance.

Method	Neigh	Sph	Ros	Gri30	Rast	Gri10	Sch
PSON	vneumm	2.62e-13	55.28001	0.004797	**18.55624**	0.014131	**0.000863**
PSOWN	vneumm	**3.24e-15**	**41.07052**	**0.001905**	19.45144	**0.009542**	0.002397
PSOWN	slbest	1.99e-14	56.63672	0.006164	67.66419	0.021161	0.001492
PSO	vneumm	2.92e-06	112.8217	0.012733	66.43264	0.049096	0.001946
PSO	lbest	0.000262	129.2118	0.009787	80.90703	0.046919	0.001945
PSON	slbest	3.11e-13	55.18398	0.009572	90.0651	0.043936	0.001952
PSO	svneumm	3.14e-08	93.4288	0.014248	74.79059	0.066009	0.002916
PSO	gbest	1.34e-09	77.63808	0.030719	70.24394	0.08057	0.004615
PSO	slbest	7.34e-06	117.461	0.010339	84.50814	0.051867	0.005105
PSO	sgbest	1.09e-08	129.6299	0.081115	78.22841	0.094794	0.004372
PSOWN	lbest	36.50434	3584.292	1.377846	182.3623	0.152637	0.001006
PSON	lbest	60.28522	12801.69	1.552355	178.3093	0.172	0.001592
PSON	sgbest	777.2214	66402.33	15.4314	106.8297	0.239283	0.007165
PSOWN	gbest	1197.985	104401.3	14.79815	85.97243	0.165539	0.00785
PSOWN	sgbest	1244.63	105420.1	14.325	103.51	0.190038	0.006315
PSON	gbest	1106.068	81166.54	16.07105	78.22476	0.259984	0.008197

Table 3. Comparison of the average performance of the configurations over the functions

Both PSON and PSOWN present better results than the canonical PSO. The best result obtained by the canonical PSO is with the *von Neumann* sociometry, ranking 7th in proportion rank and 10th in performance. Specifically, PSON and PSOWN are able to converge in the Griewank function in 10 dimensions, while the canonical PSO couldn't. Table 3 compares some of the best performing configurations for each algorithm.

It is interesting to notice that PSON is able to achieve better results than PSOWN (specifically, with the *von Newmann* neighborhood), even though the latter uses more problem specific information; it does not help to weight the

contributions of the neighbors based on their performance. Even if this conclusion is somewhat counterintuitive, it is a bringer of good news as the simpler PSON has a lower time complexity than PSOWN.

5 Conclusions and further work

Our proposed approach achieves a better performance in fewer iterations (and thus using fewer function evaluations) for the test functions. We conclude that there is valuable information to be gained by combining aspects of each of the individuals from the neighborhood and not only from the best individual. It is interesting to notice that the best results do not take into consideration the past experience of the individual.

It is important to study the influence of topologies in these two variants. Even though the *von Neumann* sociometry seems to be a very good performer, it is important to understand why it is such a good choice. It seems also important to consider the implications of apparently not needing to consider the past experience of the individual. Also, these new variants need to be thoroughly tested with different problems, to attest for their robustness.

References

1. M. Clerc and J. Kennedy. The particle swarm: Explosion, stability, and convergence in a multi-dimensional complex space. *IEEE Transactions on Evolutionary Computation*, pages 58–73, Feb 2002.
2. J. Kennedy. Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance. In *Proceedings of the 1999 Conference on Evolutionary Computation*, pages 1931–1938. IEEE Computer Society, 1999.
3. J. Kennedy and R. Mendes. Topological Structure and Particle Swarm Performance. In *To appear in* Congress on Evolutionary Computatiton (CEC2002), Honululu, Hawaii, May 2002. IEEE Computer Society.
4. E. Ozcan and C. Mohan. Particle swarm optimization: surfing the waves. In *Proceedings of the 1999 Congress on Evolutionary Computation*, pages 1939–1944, Piscataway, NJ, 1999. IEEE Service Center.
5. R. G. Reynolds and C. Chung. Knowledge-based self-adaptation in evolutionary programming using cultural algorithms. In *Proceedings of IEEE International Conference on Evolutionary Computation (ICEC'97)*, pages 71–76, 1997.
6. Y. Shi and R. Eberhart. Parameter selection in particle swarm optimization. In *Proceedings of the Seventh Annual Conference on Evolutionary Programming*, Mar 1998.
7. David H. Wolpert and William G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.
8. David H. Wolpert and William G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, April 1997.