Université Libre de Bruxelles

Institut de Recherches Interdisciplinaires
et de Développements en Intelligence Artificielle

A Comparison of Particle Swarm Optimization Algorithms Based on Run-Length Distributions

Marco A. Montes de Oca, Thomas Stützle,
Mauro Birattari and Marco Dorigo

IRIDIA – Technical Report Series

Technical Report No.
TR/IRIDIA/2006-013

April 2006

IRIDIA – Technical Report Series
ISSN 1781-3794

Published by:

IRIDIA, *Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle*
Université Libre de Bruxelles
Av F. D. Roosevelt 50, CP 194/6
1050 Bruxelles, Belgium

Technical report number TR/IRIDIA/2006-013

Revision history:

TR/IRIDIA/2006-013.001 April 2006

A Comparison of Particle Swarm Optimization Algorithms Based on Run-Length Distributions

Marco A. Montes de Oca mmontes@ulb.ac.be
Thomas Stützle stuetzle@ulb.ac.be
Mauro Birattari mbiro@ulb.ac.be
Marco Dorigo mdorigo@ulb.ac.be
IRIDIA, Université Libre de Bruxelles, Brussels, Belgium

April 2006

Abstract

In this paper we report an empirical comparison based on run-length distributions (RLDs) of some of the most influential Particle Swarm Optimization (PSO) algorithms. The advantage of our approach over the usual report pattern (average iterations to reach a predefined goal, success rates, and standard deviations) found in the current PSO literature is that it is possible to evaluate the performance of an algorithm on different application scenarios at the same time. The RLDs reported in this paper show some of the strengths and weaknesses of the studied algorithms and suggest ways of improving their performance.

1 Introduction

Since the introduction of the first Particle Swarm Optimization (PSO) algorithm by Kennedy and Eberhart [2, 9], many variants of the original algorithm have been proposed. The approach followed by many researchers to evaluate the performance of their variants had been to compare the proposed variant with the original version or, more recently, with the so-called canonical version [1]. In many cases, these new variants are reported to perform better, see for instance [5, 8, 10, 12].

Unfortunately, since there are no cross-comparisons among variants, there is no general agreement on which PSO variant(s) could be considered the state-of-the-art in the field. The motivation of conducting the comparison reported in this paper, was the identification of these variant(s). However, determining the state-of the-art algorithm is not a trivial task. In particular, one must be aware of the possible application scenarios in which a stochastic optimization algorithm may be used. In general, there is a utility value associated with a solution of a certain quality depending on how long an algorithm takes to get it [6]: if two optimization algorithms return the same solution, the faster one should be preferred.

The run-time of a stochastic optimization algorithm is a random variable and its distribution is known as the run-time distribution. However, since in continuous optimization it is common to measure run-times in terms of the number of function evaluations needed to achieve a certain solution quality, we talk of run-length distributions (RLDs) rather than run-time distributions. This is the approach followed in this paper.

An RLD completely characterizes the behavior of a stochastic optimization algorithm on a particular problem, regardless of the actual application scenario in which we may be interested in. This is the main reason why we chose to evaluate some of the most influential PSO algorithms using RLDs. As a bonus, an analysis based on RLDs allows the identification of some strengths and weaknesses of the studied algorithms and may also be used to design improved versions.

The rest of the paper is organized as follows. Section 2 briefly describes the PSO technique, the criteria we used to select the variants compared in this paper and the selected variants themselves.

Section 3 describes the experimental setup adopted for our comparison. Section 4 presents the development of the solution quality over time and the RLDs of the studied algorithms. Section 5 sums up the main contributions and results presented in the paper.

2 Particle Swarm Optimization Algorithms

In the original PSO algorithm [2,9], a fixed number of solutions (called *particles* in a PSO context) are randomly located in a d-dimensional solution space. A particle i at time step t has a position vector $\vec{x}_i^t$ and a velocity vector $\vec{v}_i^t$, which are also randomly initialized. A fitness function $f : S \rightarrow \Re$ where $S \subset \Re^d$, determines the quality of a particle's position, i.e., a particle's position represents a solution to the problem being solved. Each particle i has a vector $\vec{p}_i$ that represents its own best previous position that has an associated fitness value $pbest_i = f(\vec{p}_i)$. Finally, the best position the swarm has ever visited during a run is stored in a vector $\vec{s}$ whose fitness value is $gbest = f(\vec{s})$.

The algorithm iterates updating the velocities and positions of the particles until a stopping criterion is met. The update rules are:

$$\vec{v}_i^{t+1} = \vec{v}_i^t + \varphi_1 \vec{U}_1(0,1) * (\vec{p}_i - \vec{x}_i^t) + \varphi_2 \vec{U}_2(0,1) * (\vec{s} - \vec{x}_i^t) \,, \tag{1}$$

$$\vec{x}_i^{t+1} = \vec{x}_i^t + \vec{v}_i^{t+1} \,, \tag{2}$$

where φ_1 and φ_2 are two constants called the *cognitive* and *social* acceleration coefficients respectively, $\vec{U}_1(0,1)$ and $\vec{U}_2(0,1)$ are two d-dimensional uniformly distributed random vectors in which each component goes from zero to one, and $*$ is an element-by-element vector multiplication operator.

The variants we compare in this study were selected either because they are among the most commonly used in the field or because they look very promising. They go from trying different inertia weight updating schemes to the use of dynamic neighborhood topologies. In the following subsections, we describe in more detail the selected variants.

2.1 Canonical Particle Swarm Optimizer

Clerc and Kennedy [1] introduced the constriction factor in PSO to control the convergence properties of the particles. This constriction factor is added in Equation 1 giving

$$\vec{v}_i^{t+1} = \chi \left(\vec{v}_i^t + \varphi_1 \vec{U}_1(0,1) * (\vec{p}_i - \vec{x}_i^t) + \varphi_2 \vec{U}_2(0,1) * (\vec{s} - \vec{x}_i^t) \right) \,, \tag{3}$$

with

$$\chi = 2k / \left(\left| 2 - \varphi - \sqrt{\varphi^2 - 4\varphi} \right| \right) \,, \tag{4}$$

where $k \in [0,1]$, $\varphi = \varphi_1 + \varphi_2$ and $\varphi > 4$. Usually, χ is set to 0.729 and φ_1 and φ_2 are both set to 2.05 [3, 16]. This variant has been so widely used that it is known as the *canonical* PSO.

2.2 Time-Varying Inertia Weight Particle Swarm Optimizer

Shi and Eberhart [12, 14] introduced the idea of a time-varying inertia weight. The idea was to control the diversification–intensification behavior of the original PSO. The velocity update rule is

$$\vec{v}_i^{t+1} = w(t)\vec{v}_i^t + \varphi_1 \vec{U}_1(0,1) * (\vec{p}_i - \vec{x}_i^t) + \varphi_2 \vec{U}_2(0,1) * (\vec{s} - \vec{x}_i^t) \,, \tag{5}$$

where $w(t)$ is the time-varying inertia weight which usually is linearly adapted from an initial value to a final one. In most cases, φ_1 and φ_2 are both set to 2.

There are two ways of varying the inertia weight in time: decreasingly (e.g., as in [12–14]) and increasingly (e.g., as in [17, 18]). In this paper, we included both variants for the sake of completeness. Normally, the starting value of inertia weight is set to 0.9 and the final to 0.4 linearly decreasing in time. Zheng et al. [17, 18], use the opposite settings. In the results section, these variants are identified by Dec-IW and Inc-IW, respectively.

2.3 Stochastic Inertia Weight Particle Swarm Optimizer

Eberhart and Shi [4] proposed another inertia weight variation approach in which it is randomly selected according to a uniform distribution in the range [0.5,1.0]. This range was inspired by Clerc and Kennedy's constriction factor. In this version, the acceleration coefficients are set to 1.494 just as in the canonical PSO. Although this variant was originally proposed for dynamic environments, it has also been shown to be a competitive optimizer for static ones [11]. In the results section this variant is identified by Sto-IW.

2.4 Fully Informed Particle Swarm Optimizer

In the fully informed particle swarm (FIPS) proposed by Mendes et al. [10], a particle uses information from all its topological neighbors. This variant is based on the fact that Clerc and Kennedy's constriction factor does not enforce that the value φ should be split only between two attractors.

For a given particle, the way φ (i.e., the sum of the acceleration coefficients) is decomposed is $\varphi_k = \frac{\varphi}{|\mathcal{N}|}\ \forall k \in \mathcal{N}$ where $\mathcal{N}$ is the neighborhood of the particle. As a result, the new velocity update equation becomes

$$\vec{v}_i^{t+1} = \chi \left[\vec{v}_i^t + \sum_{k \in \mathcal{N}} \varphi_k \mathcal{W}(k) \vec{U}_k(0,1) * (\vec{p}_k - \vec{x}_i^t) \right] , \quad (6)$$

where $\mathcal{W}(k)$ is a weighting function.

2.5 Self-Organizing Hierarchical Particle Swarm Optimizer with Time-varying Acceleration Coefficients

The self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients (HPSOTVAC) proposed by Ratnaweera et al. [11] drops the velocity term from the right side of Equation 5. If a particle's new velocity becomes zero (in any dimension), it is reinitialized to some value proportional to the maximum allowable velocity V_{max}. HPSOTVAC linearly adapts the value of the acceleration coefficients φ_1 and φ_2 to enforce the diversification behavior at the beginning of the run and the intensification behavior at the end. φ_1 is decreased from 2.5 to 0.5 and φ_2 varies from 0.5 to 2.5. Finally, the reinitialization velocity is also linearly decreased from V_{max} at the beginning of the run to $0.1\,V_{max}$ at the end.

2.6 Adaptive Hierarchical Particle Swarm Optimizer

Proposed by Janson and Middendorf [7], the adaptive hierarchical PSO (AHPSO) is an example of a PSO with dynamic adaptation of the population topology. In AHPSO, the topology is a tree-like structure in which particles with a higher fitness evaluation are located in the upper nodes of the tree. At each iteration, a child particle updates its velocity considering its own previous best performance and the previous best performance of its parent. Additionally, before the velocity updating process takes place, the previous best fitness value of any particle is compared with that of its parent. If it is better, child and parent swap their positions in the hierarchy.

The branching degree of the tree is a factor that can balance the diversification–intensification behavior of the algorithm. To dynamically adapt the algorithm to the stage of the optimization process, the branching degree is decreased k_{adapt} degrees (one at a time) until a certain minimum degree d_{min} is reached. This process takes place every certain number of iterationsf_{adapt}. The parameters that control this process need to be tuned for each problem [7]. In our experiments, for the reasons explained in the next section, we set the initial branching factor to 20, parameters d_{min}, f_{adapt}, and k_{adapt} were set to 2, $1000 * m$, and 3 respectively, where m is the number of particles.

3 Experimental Setup

All the PSO variants described in the previous section were implemented for this comparison. To ensure the correctness of our implementations, we tested them on the same problems with the same parameters as reported in the literature[1]. To allow the comparison of the results with previous works, we used some of the most common benchmark functions in the PSO literature: Sphere, Rosenbrock, Rastrigin, Griewank, and Schaffer's F6 functions in 30 dimensions. The mathematical definition of these functions is readily available in the literature (see, e.g., [16]). In our runs, these functions were shifted and biased as in [15][2]. Because of this, our initializations are, in all cases, asymmetric with respect to the global optimum.

The results reported here are based on 100 independent trials each running for 1 000 000 function evaluations. In our experiments, we used swarms of 20 particles using two different topologies: fully connected and ring. The results are organized by population topology. Both topologies included self-references (i.e., every particle is a neighbor to itself). This separation was needed to highlight the influence of the used topology in the behavior of the algorithms. Note that the AHPSO algorithm uses neither a fully connected topology nor a ring topology and therefore appears in both sets of results.

Before proceeding to the presentation of our results, it is worth noting that most PSO algorithms are not robust in their parameterization. For example, in the PSO variants based on a time-varying inertia weight, the slope of the increasing or decreasing inertia weight function is determined by the maximum number of function evaluations. Another problem (for comparison purposes) is that it is also possible to fine-tune the parameters of a variant to solve a particular problem. A possible solution to this problem is to fine-tune all variants for the problem at hand and proceed with the comparison; however, if our aim is to solve real-world problems which generally have a structure we do not know in advance, we need algorithms with a set of "normally good" parameters. For this reason, in this study each algorithm used the same parameterization across the benchmark problems. The actual values chosen for the parameters have already been mentioned in the preceding sections.

4 Results

Tables 1 and 2 show the average value and standard deviation of the number of function evaluations needed to achieve a certain solution quality with fully connected and ring topologies, respectively. For each function, there are three different solution qualities. The first one corresponds to the usual goal for that function (see, e.g., [16]). The second and third can be considered medium and high qualities, respectively.

Most variants, most notably FIPS, are greatly affected in their performance by the used topology. With a fully connected topology, most of the tested variants reach the specified solution quality faster than with the ring topology. FIPS performs poorly with this topology: only in 4 out of 15 cases it reaches the specified solution quality. However, whenever it does, it is the fastest algorithm.

With the Rastrigin function, HPSOTVAC is the only variant in reaching the highest solution quality. The Rastrigin function is a highly multimodal function and reaching this high solution quality level, is an indication of how well an algorithm is capable of escaping from local minima. Apparently, HPSOTVAC succeeds at this but at the cost of a greater number of function evaluations than other variants.

When all other variants use the ring topology, AHPSO performs better that it did when the others used a fully connected topology. This is expected since AHPSO adapts the population hierarchy from a highly connected one to a loosely connected one, so it exploits the benefits of

[1] For space restrictions, we refer the interested reader to the following address: http://iridia.ulb.ac.be/∼mmontes/performanceANTS06.html

[2] The values of the optima are specified in Tables 1 and 2.

Table 1: Average value and standard deviation of the number of function evaluations needed to achieve a certain solution quality using the fully connected topology. (Only successful runs are considered) {f1=Griewank (optimum at -130.0), f2=Rastrigin (optimum at -330), f3=Rosenbrock (optimum at 390), f4=Schaffer's F6 (optimum at -300), f5=Sphere (optimum at -450)}

Function	S.Q.	AHPSO	Canonical	FIPS	Dec-IW	HPSOTVAC	Inc-IW	Sto-IW
f1	0.077%	9641.28 1413.02	**8345.78** 1271.42	– –	433036 24012.3	25741.8 1647.09	8365.53 852.798	9713.76 1483.53
	0.01%	11613.8 1508.08	**9784.1** 1153.03	– –	442995 15996.2	34474.3 7430.55	10115.8 1274.6	11806.7 1788.94
	0.001%	12994.3 1481.04	**11322.7** 2120.16	– –	453478 21515.5	57830 61113.8	41082.6 166025	13306.2 2335.77
f2	30.30%	4010.94 1478.62	3836.62 1065.18	**960** 56.5685	364350 42590.1	29852.8 21636.7	53104.4 212169	4820.73 1534.74
	15%	6295 1123.02	**5550** 1400.07	– –	427895 30835.2	101322 44487.3	516798 490241	7758 2554.59
	1%	– –	– –	– –	– –	**635060** 133964	– –	– –
f3	25.64%	108546 150900	50148.6 58874	– –	492989 70214.1	489920 258767	**22381.2** 19505.5	56747.8 105101
	10%	153596 204835	87750.5 90431.8	– –	548089 85071.9	665170 205934	**67398.8** 104077	95153.8 148441
	1%	353182 201800	**168799** 97989.6	– –	653042 116809	762488 218223	288477 208034	232858 152029
f4	3.3×10^{-6}%	33636.3 93014.2	21044.6 44560.4	**5230** 3921.31	115876 24171.4	84893.3 184640	72287.1 196370	56837.6 172404
	1×10^{-7}%	34306.1 93024.3	21478.2 44536.7	**5540** 3964.94	131253 22963.3	86826 184097	58508.4 161070	57350.8 172359
	1×10^{-8}%	34540 92995.1	21789.6 44506.6	**5724** 3932.08	140434 19112.8	88095.1 183701	58760.6 161142	57698.4 172325
f5	0.0022%	11342.8 1386.8	10913 2255.69	– –	433345 6885.29	31210.4 1436.41	**8135.15** 944.406	11127.8 1317.03
	0.0002%	14000.6 1468.18	13122.4 2648.04	– –	446824 6707.75	40913 1616.75	**9772.04** 946.009	13680.4 1919.03
	0.00001%	15862.6 1506.09	14955.8 2795.72	– –	456062 6346.54	48546.6 1723.04	**11147.4** 1151.96	15469.6 1890.76

Table 2: Average value and standard deviation of the number of function evaluations needed to achieve a certain solution quality using the ring topology. (Only successful runs are considered) {f1=Griewank (optimum at -130.0), f2=Rastrigin (optimum at -330), f3=Rosenbrock (optimum at 390), f4=Schaffer's F6 (optimum at -300), f5=Sphere (optimum at -450)}

Function	S.Q.	AHPSO	Canonical	FIPS	Dec-IW	HPSOTVAC	Inc-IW	Sto-IW
f1	0.077%	9641.28 1413.02	12377.8 849.032	**8030.2** 957.811	456568 11431.9	29862 1538.57	17268.8 1192.31	15530.2 1538.17
	0.01%	**11613.8** 1508.08	15865.6 4322.18	12454.5 7074.66	499384 68132.2	40342.2 3660.7	29852.1 50030.8	23091.9 37164.4
	0.001%	**12994.3** 1481.04	32157.9 62991	21422.5 26631.3	536619 90978.1	60952.1 36481	61165.4 115035	35051.8 71418.5
f2	30.30%	**4010.94** 1478.62	30091.3 89696.1	22599.6 9998.23	360126 63803.9	30052.8 12946.6	34257.8 142877	19767.8 84546.4
	15%	**6295** 1123.02	– –	136648 100821	460690 59069.9	117057 37364.8	685132 429521	206669 275732
	1%	– –	– –	– –	– –	**811247** 107569	– –	– –
f3	25.64%	108546 150900	**104684** 130114	226828 252722	518732 70783.1	361997 325196	127764 154960	151060 188680
	10%	**153596** 204835	189872 216622	320153 270742	605140 124911	291822 263481	173475 166956	215458 233277
	1%	**353182** 201800	426285 221124	443895 242211	734466 146611	– –	531935 270172	458950 262995
f4	3.3×10^{-6}%	33636.3 93014.2	43691.4 89702.5	40416.9 90967.5	123649 28220.1	126988 204489	**28014** 32195.1	40241.2 79386.6
	1×10^{-7}%	34306.1 93024.3	44897.6 89965.7	49353.5 92344.4	139871 27231.6	129022 204003	**29321.8** 32216.9	42340.2 82079
	1×10^{-8}%	34540 92995.1	45482.8 89856.7	54939.6 92882.9	150797 27877.7	130576 203686	**29771.6** 32327.5	43142.4 82122.1
f5	0.0022%	11342.8 1386.8	13693.6 572.274	**8266** 480.341	459971 9304.42	35518.2 1382.3	14923 894.808	17148.6 1129.21
	0.0001%	14000.6 1468.18	16421.4 663.345	**9920.8** 491.61	477006 8254.22	47480 1469.75	18077.4 956.283	20660.8 1265.5
	0.00001%	15862.6 1506.09	18484.2 684.266	**11169.6** 523.234	488408 7977.23	56889 1923.1	20430.6 1055	23289.4 1344.88

converging faster at the beginning of the run. As seen from the results, fast convergence is somehow associated to the fully connected topology, or at least with a highly connected one.

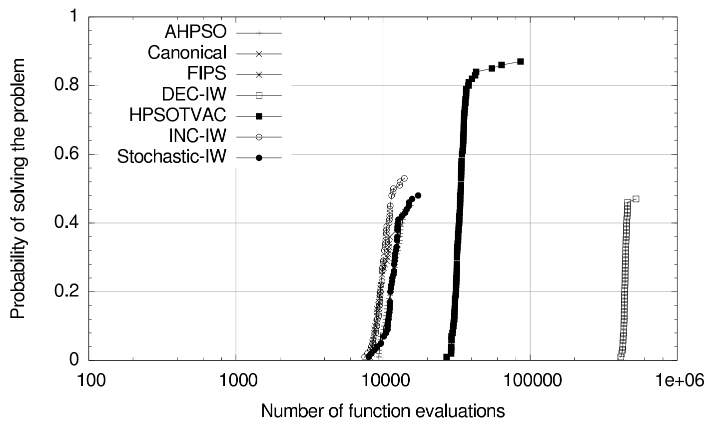
The Canonical PSO and the Increasing Inertia Weight variants are not bad at all. With the fully connected topology, they are the best performers in 10 out of 15 cases. With the ring topology, this number drops to only 4. These variants clearly exploit the convergence properties of the fully connected topology.

The data shown in the Tables 1 and 2 should be taken *cum grano salis*. The averages and standard deviations reported there are computed over successful runs only. This could be potentially misleading if the probability of achieving those results is not also reported. RLDs are the empirical cumulative probabilities of finding a solution of certain quality. Tables 3 and 4 show the RLDs in four benchmark functions[3]. The shown RLDs correspond to solution qualities of 0.01% for Griewank function, 30% for Rastrigin function, 10% for Rosenbrock function, and 0.0001% for Schaffer's F6 function. The results are organized by population topology: on the left, the results obtained using a fully connected topology; on the right, using the ring topology.

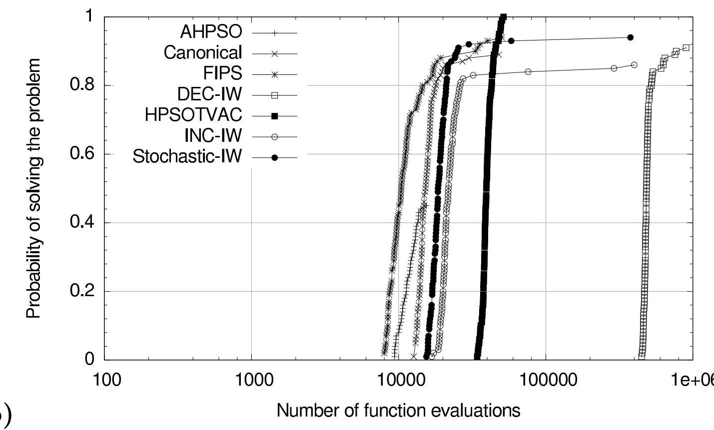
The "slope" of the shown curves point out interesting features of the algorithms. If an RLD for a given solution quality is steep (but complete), it means that the algorithm finds the solution easily. Of course, if the demanded quality is high, the algorithm will need more function evaluations to find it, but this will only affect its position in RLD graphs (the higher the quality, the more to the right), not its slope. This is the case with the HPSOTVAC variant using the ring topology in the Griewank function (Table 3, (b)). It can be seen, however, that this is an exception and not the rule. Most variants have curves with low steepness or steep incomplete curves.

The phenomenon that explains this behavior is *stagnation*. If an algorithm has a steep incomplete RLD, a restarting mechanism could greatly help. Such an algorithm gets stuck from time to time, but since it finds a solution relatively easy, the most straightforward way to solve the problem is to restart the search. Almost all variants show this behavior when solving Griewank and Rastrigin problems.

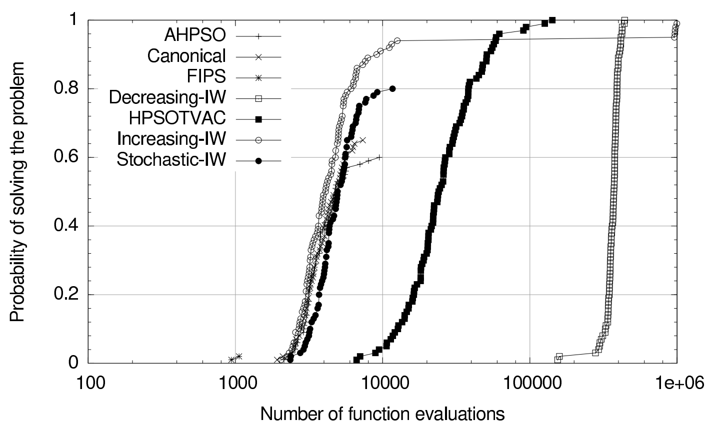
[3]The RLDs of the algorithms in the Sphere function can be found in the already mentioned web address.



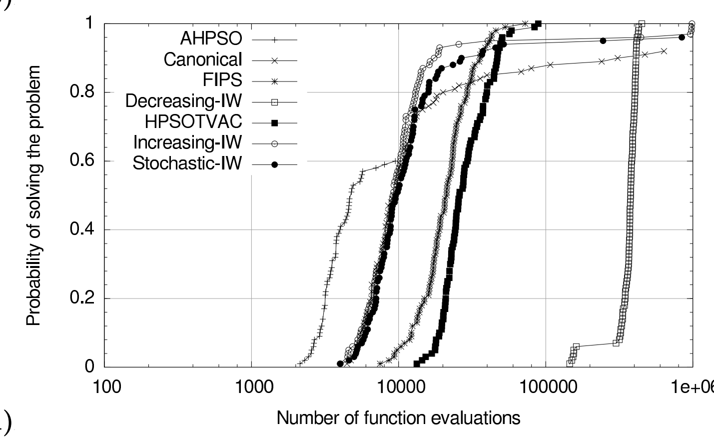
(a)



(b)



(c)



(d)

Table 3: Run-Length Distributions. (a) and (b), the results obtained with Griewank function. (c) and (d), the ones with Rastrigin

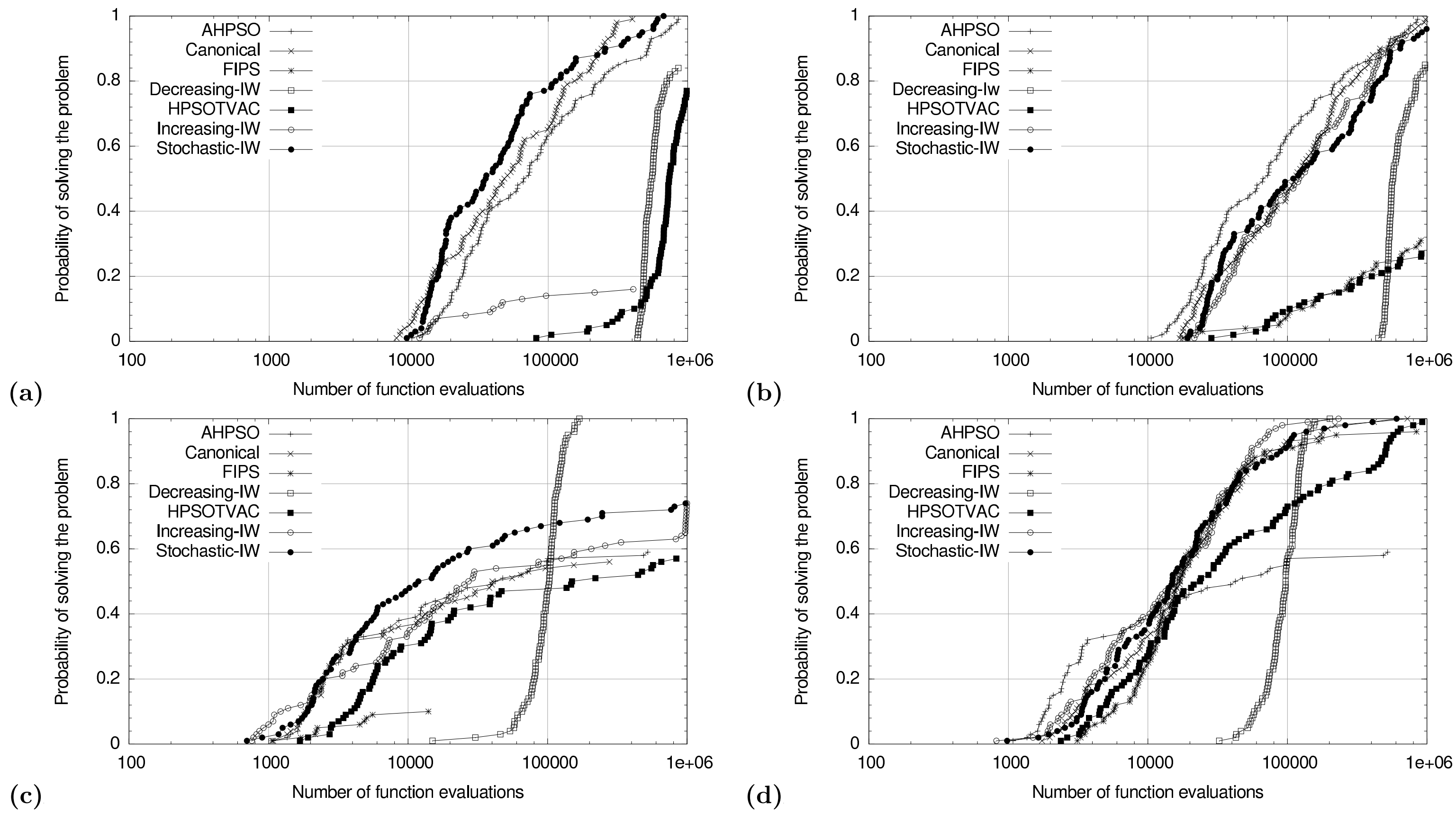


Table 4: Run-Length Distributions. (a) and (b), the results obtained with Rosenbrock function. (c) and (d), the ones with Schaffer's F6

A different problem can be seen in the RLDs for Rosenbrock and Schaffer's F6 functions. In these cases, the RLDs have a low steepness which highlights the lack of diversification strategies in most of the algorithms.

The only variant that do not follow the pattern in these two problems is the one based on a decreasing inertia weight and is the only one designed with diversification in mind. This variant was designed to explore the search space at the beginning and intensify the search near the end of a run. This could explain the steepness of its RLDs in these two problems.

5 Conclusions

In this paper we empirically compared seven of the most influential or promising variants of the original particle swarm optimization algorithm. Our approach was to use Run-Length Distributions (RLDs) and statistics of the solution quality development over time.

Regarding the behavior shown by the tested PSO variants, it is evident how important is the choice of the neighborhood topology in the performance of PSO algorithms. This is something already known in the field, but the measurement of its influence in the stagnation behavior of PSO algorithms had never been done before. With respect to our initial motivation, we limited ourselves to the comparison of some of the most influential variants, and from our results we did not find any dominant variant.

One of the advantages of RLDs is that they allow the evaluation of a stochastic optimization algorithm regardless of the actual application scenario it may be used in. Another advantage is that they allow the identification of some strengths and weaknesses of the studied algorithms that can be used to improve their performance. Future research will focus on exploiting the information provided by RLDs to the *engineering* of PSO variants. We sketched how this could be done.

Acknowledgments. This work was supported by the ANTS project, an *Action de Recherche Concertée* funded by the Scientific Research Directorate of the French Community of Belgium. Marco Montes de Oca acknowledges support from the Programme Alβan, the European Union Programme of High Level Scholarships for Latin America, scholarship No. E05D054889MX. Thomas Stützle and Marco Dorigo acknowledge support from the Belgian National Fund for Scientific Research (FNRS), of which they are a Research Associate and a Research Director, respectively.

References

[1] M. Clerc and J. Kennedy. The particle swarm–explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[2] R. Eberhart and J. Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, pages 39–43, Piscataway, NJ, 1995. IEEE Press.

[3] R. Eberhart and Y. Shi. Comparing inertia weights and constriction factors in particle swarm optimization. In *Proceedings of the 2000 IEEE Congress on Evolutionary Computation*, pages 84–88, Piscataway, NJ, 2000. IEEE Press.

[4] R. Eberhart and Y. Shi. Tracking and optimizing dynamic systems with particle swarms. In *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, pages 94–100, Piscataway, NJ, 2001. IEEE Press.

[5] H. Fan. A modification to particle swarm optimization algorithm. *Engineering Computations*, 19(8):970–989, 2002.

[6] H. H. Hoos and T. Stützle. *Stochastic Local Search: Foundations and Applications.* Morgan Kaufmann Publishers, San Francisco, CA, USA, 2004.

[7] S. Janson and M. Middendorf. A hierarchical particle swarm optimizer and its adaptive variant. *IEEE Transactions on Systems, Man and Cybernetics–Part B*, 35(6):1272–1282, 2005.

[8] J. Kennedy. Stereotyping: Improving particle swarm performance with cluster analysis. In *Proceedings of the 2000 IEEE Congress on Evolutionary Computation*, pages 1507–1512, Piscataway, NJ, 2000. IEEE Press.

[9] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of IEEE International Conference on Neural Networks*, pages 1942–1948, Piscataway, NJ, 1995. IEEE Press.

[10] R. Mendes, J. Kennedy, and J. Neves. The fully informed particle swarm: Simpler, maybe better. *IEEE Transactions on Evolutionary Computation*, 8(3):204–210, 2004.

[11] A. Ratnaweera, S. K. Halgamuge, and H. C. Watson. Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. *IEEE Transactions on Evolutionary Computation*, 8(3):240–255, 2004.

[12] Y. Shi and R. Eberhart. A modified particle swarm optimizer. In *Proceedings of the 1998 IEEE World Congress on Computational Intelligence*, pages 69–73, Piscataway, NJ, 1998. IEEE Press.

[13] Y. Shi and R. Eberhart. Parameter selection in particle swarm optimization. In *Proceedings of the 7th International Conference on Evolutionary Programming VII*, pages 591–600. LNCS Vol. 1447. Springer-Verlag, New York, 1998.

[14] Y. Shi and R. Eberhart. Empirical study of particle swarm optimization. In *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, pages 1945–1950, Piscataway, NJ, 1999. IEEE Press.

[15] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical Report 2005005, Nanyang Technological University, Singapore and IIT Kanpur, India, May 2005.

[16] I. C. Trelea. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Information Processing Letters*, 85(6):317–325, 2003.

[17] Y.-L. Zheng, L.-H. Ma, L.-Y. Zhang, and J.-X. Qian. Empirical study of particle swarm optimizer with an increasing inertia weight. In *Proceedings of the 2003 IEEE Congress on Evolutionary Computation*, pages 221–226, Piscataway, NJ, 2003. IEEE Press.

[18] Y.-L. Zheng, L.-H. Ma, L.-Y. Zhang, and J.-X. Qian. On the convergence analysis and parameter selection in particle swarm optimization. In *Proceedings of the 2003 IEEE International Conference on Machine Learning and Cybernetics*, pages 1802–1807, Piscataway, NJ, 2003. IEEE Press.