The Effect of Probability Distributions on the Performance of Quantum Particle Swarm Optimization for Solving Dynamic Optimization Problems

Kyle Robert Harrison
Department of Computer Science
University of Pretoria
Pretoria, South Africa
krharrison28@gmail.com

Beatrice M. Ombuki-Berman
Department of Computer Science
Brock University
St. Catharines, ON, Canada
bombuki@brocku.ca

Andries P. Engelbrecht
Department of Computer Science
University of Pretoria
Pretoria, South Africa
engel@cs.up.ac.za

***Abstract*—The quantum particle swarm optimization (QPSO) algorithm was developed to address the limitations of the traditional particle swarm optimization (PSO) algorithm in dynamic environments. Some particles in the QPSO algorithm are chosen as "quantum" particles, and the positions of these are sampled uniformly within a radius (i.e., a hypersphere) centred around the global best particle. The remainder of particles follow standard PSO behaviour. This paper proposes sampling various alternative probability distributions to update the positions of quantum particles. Ten probability distributions are examined on dynamic environments with varying dimensionalities, temporal change severities, and spatial change severities, with both single-peak and five-peak environments considered. Results indicated that the most effective distribution to use is dependent upon the type of dynamism present. In general, it was observed that a small quantum radius was preferable to a large radius, indicating that exploitation is more beneficial than exploration with regards to QPSO performance. Finally, despite having been commonly used in various QPSO applications, the performance of the uniform distribution was found to be sub-par.**

I. Introduction

Many real-world optimization problems exhibit dynamic behaviour where the optimal solution varies over time. For example, environmental changes may shift the position of the optima, the optimal fitness value, or both. Such dynamic problems are more challenging than their static counterparts – simply locating the optimal solution is no longer sufficient as the position of the optimum may change over time. Moreover, extreme cases of dynamism (i.e., ones where environmental changes are very frequent) are akin to random environments in the sense that little to no environmental knowledge can be gathered before being (potentially) invalidated by a change in the problem landscape. Similarly, dynamic problems with less frequent changes also pose a problem to optimizers which exhibit convergence, as the limited diversity in such a scenario may hinder an adequate response to an environmental change.

The particle swarm optimization (PSO) algorithm [1] is a stochastic optimization technique based on the swarming behaviour found in flocks of birds. The PSO algorithm iteratively calculates and applies a velocity vector to the position of each agent, causing them to have flight-like movements. Agents in the PSO algorithm, known as particles, are attracted to two positions in the search space which govern their movement: their personal best position and the global best position found by any particle during the PSO execution. The PSO algorithm was originally proposed as a continuous-space optimizer for static environments and has found great success in a wide variety of domains [1], [2], [3]. However, both challenges previously highlighted for dynamic environments, namely knowledge being invalidated by environmental changes and lack of diversity, lead to major performance issues for PSO in dynamic environments [4].

Quantum particle swarm optimization (QPSO) [5] was introduced as a dynamic variant of PSO aimed to address its known drawbacks in dynamic environments. In the QPSO algorithm, some proportion of particles are designated as "quantum" particles and forego the traditional particle update strategy. Rather, their positions are sampled uniformly within a radius around the global best particle. This strategy is intended to inject diversity within the population, especially late in the run, as this prevents all particles from converging to a single point. While most studies employing QPSO sample the position uniformly around the global best position, the use of alternative distributions has been recognized but only the uniform, Gaussian, and non-uniform (linearly decreasing density) distributions have been examined [6]. Blackwell *et al.* [6] found that of the three examined distributions, the uniform distribution performed the best with respect to offline error. However, this experiment was small in scope, as it was tangential to the overall study, and explored only limited types of dynamism. In contrast, this paper examines ten probability distributions with a direct focus on examining a wide variety of dynamic environment types.

The quantum particles in the QPSO algorithm provide a mechanism of both exploration and exploitation. During the early phase of QPSO execution, or shortly after an environmental change, the quantum particles work as an exploitation mechanism whereby they exploit the global best position. Later during QPSO execution, or after a relatively long time has passed since an environmental change, i.e., when convergence has begun, the quantum particles provide an exploration mechanism by retaining a base level of diversity. This level of diversity is largely based on the radius of the quantum cloud. Thus, one can control the balance of exploration and exploitation by controlling the degree to which quantum particles centre around the global best position versus tending away from it. To assess the effect of probability distributions

978-1-4799-7560-0/15 $31.00 © 2015 IEEE
DOI 10.1109/SSCI.2015.44

on the performance of the QPSO algorithm, ten probability distributions are examined.

The remainder of this paper is structured as follows. Background information on dynamic environments, PSO, and QPSO is given in Section II. Section III introduces the examined probability distributions while Section IV describes the experimental procedures. Experimental results and a discussion of these results are provided in Section V. Finally, concluding remarks are given in Section VI.

II. Background

This section provides the necessary background information for the remainder of the paper.

A. Dynamic Environments

A dynamic environment is, generally speaking, an environment which changes over time. While this definition is quite vague, a great deal of research effort has been devoted to better describe and classify dynamic environments. Many of these classification schemes are based on the characteristics of the dynamism. Duhain and Engelbrecht [7] point out three major benefits to having a more complete classification of dynamic environments. Firstly, a more descriptive classification of dynamic optimization problems allows for better inferences about algorithmic strengths and weakness to be made based on their performance. Secondly, one can reason about algorithmic performance on an unseen problem once the environmental type of the problem is known. Finally, algorithms can exhibit observable and predictable characteristics when faced with certain types of dynamism. Thus, when presented with an unknown dynamic environment, observing such characteristics in an algorithm's behaviour may be indicative of the environment type, allowing one to gather previously unknown information about the environment.

Angeline [8] focused on classifying the trajectories and patterns of changes which the optimum undergoes. For example, a *linear* optimum follows a linear trajectory, a *circular* optimum follows a periodic pattern, and a *random* optimum has no discernible, or rather recognized, pattern. De Jong [9] examined the frequency and magnitude of changes, referred to as the temporal and spatial severity, respectively, to differentiate between *drifting* and *abrupt* environments. Environments of the former type change frequently but changes are small in magnitude while the latter observes infrequent but relatively large-magnitude changes. Weicker [10] defined six classes of dynamism based on a combination of characteristics from Angeline and De Jong's classification schemes. Furthermore, Weicker introduced the notion of *homogeneity* where a homogeneous environmental change is one in which all optima change in the same manner.

Eberhart *et al.* [11], [12] proposed three classes of dynamic environments based on whether the optima change in position (Type I), value (Type II), or both (Type III). Recently, Duhain and Engelbrecht [7] proposed a more generalized framework for classifying dynamic environments based on four environment classes, namely *quasi-static*, *progressive*, *abrupt*, and *chaotic*. Each of these classes differ in their relative temporal and spatial severities. On one extreme, quasi-static environments observe infrequent, small changes while the other extreme, chaotic environments, observe frequent and large changes. Progressive and abrupt environments are akin to the drifting and abrupt environments of De Jong, respectively. Note that the boundaries between the environments can not be rigorously defined [7], however, Figure 1 provides a visualization of their relative positioning with respect to the temporal and spatial severity of the environment.

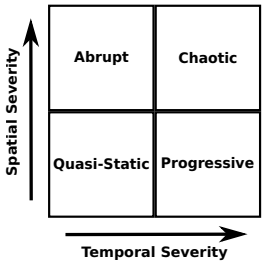


Fig. 1: Dynamic Environment Classifications

While none of the aforementioned classification schemes completely describe all aspects of dynamic environments, using a combination of the above schemes can provide a very descriptive classification of a dynamic environment. For example, by combining the classification schemes of Angeline, Weicker, Eberhart *et al.*, and the environmental classes of Duhain and Engelbrecht, 27 environmental classifications can be made which describe both the movement patterns of the optima as well as the frequency and severity of changes [7].

B. Moving Peaks Benchmark

The moving peaks benchmark (MPB) introduced by Branke [13] is a dynamic environment generator extensively used in the literature (see, for example, [4], [6], [7], [14]) due to its flexibility. MPB landscapes are constructed by defining a number of peaks consisting of a position, height, and width. The dynamism is controlled by varying the peak parameters over time. Thus, the temporal severity of the problem is defined by the change interval relative to the number of evaluations.

When an environmental change occurs, a velocity vector with fixed length s (a user-supplied parameter) is calculated for each peak. This velocity vector defines the movement of the peak, while user-supplied height and width severity parameters define scaling coefficients for the peak height and width, respectively. If a velocity vector happens to take a peak outside of the bounds of the problem, components of the velocity vector are negated as necessary to ensure the peak stays within the bounds of the problem instance. The velocity vector calculation also employs the use of an inertia coefficient, $\lambda \in [0, 1]$, to influence the movement direction. When $\lambda = 0$, the peaks will always move in random directions and when $\lambda = 1$, the peaks will always move in the same direction, randomly determined at the start of the simulation.

C. Particle Swarm Optimization

Introduced by Kennedy and Eberhart [1], the PSO algorithm was inspired by a simple model of the social dynamics of a flock of birds. The PSO algorithm is a population-based, stochastic optimization algorithm targeted towards real-valued,

static problems. The movement of each agent (referred to as a particle) is premised on two simple behaviours: move towards the best position in a neighbourhood and move towards its own personal best position.

The movement of each particle through the search space is governed by the iterative process of calculating and applying a velocity vector to the current position of the particle. The velocity for a particle is calculated as

$$\begin{aligned} v_{ij}(t+1) = \omega v_{ij}(t) + c_1 r_{1j}(t)(y_{ij}(t) - x_{ij}(t)) \\ + c_2 r_{2j}(t)(\hat{y}_j(t) - x_{ij}(t)) \end{aligned} \quad (1)$$

where $v_{ij}(t)$ is the velocity of particle i in dimension j at time t. The position of the particle in dimension j is given by x_{ij}, ω denotes the inertia coefficient, while c_1 and c_2 represent the cognitive and social coefficients, respectively. A stochastic component is added via the random constants, r_{1j} and $r_{2j} \sim U(0,1)$. Finally, $y_{ij}(t)$ and $\hat{y}_j(t)$ denote the personal and neighbourhood best positions in dimension j, respectively. Particle positions are then updated according to

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1). \quad (2)$$

Originally proposed for static environments, the PSO algorithm is known to have two major drawbacks when employed in dynamic environments, namely outdated memory and diversity loss [4]. The outdated memory issue refers to when a particles fitness and personal best position may be misleading, or incorrect, due to an environmental change. The other issue, diversity loss, occurs when the particles have exhibited convergence and all particles have converged to the same region. In a dynamic environment, lack of diversity may prevent an adequate response to an environmental change as the low velocities of the particles may inhibit the optimum tracking capabilities of the PSO algorithm.

D. Quantum Particle Swarm Optimization

Quantum PSO [5] was introduced as a variant of PSO based on a quantum model of an atom where electron positions orbit a nucleus in a non-deterministic fashion. A proportion of particles in the QPSO algorithm are designated as "quantum" particles, and these particles do not update their positions using a velocity vector. Rather, the positions of quantum particles are sampled from a probability distribution centred around the global best position. That is,

$$x_i(t+1) = d(\hat{y}(t), r_{cloud}) \quad (3)$$

where d is some probability distribution and r_{cloud} is the quantum radius. This position update strategy is analogous to the non-deterministic electron positions in the atom analogy. The remainder of particles, referred to as neutral particles, update their positions using the standard position update as given in Equation (2).

Blackwell *et al.* [6] examined three probability distributions to sample quantum particle positions from, namely the uniform, Gaussian, and non-uniform probability distributions. Their results indicated that the uniform distribution outperformed both the Gaussian and non-uniform distributions. Furthermore, Blackwell *et al.* suggest, based on theoretical and empirical evidence, that using $r_{cloud} \approx s/2$ leads to good performance as this ensures that the swarm has sufficient diversity to track a peak change of severity s [4], [6]. However, the aforementioned studies only considered a limited subset of dynamic environment types, and focused largely on the number of peaks rather than the type of dynamism. Moreover, the studies made use of relatively small change severities ($s \in [1, 10]$) and correspondingly small quantum radii ($r_{cloud} \in [0, 10]$). To the best of the authors' knowledge, the study by Blackwell *et al.* is the only study which examined alternative probability distributions for QPSO.

III. Proposed Distributions

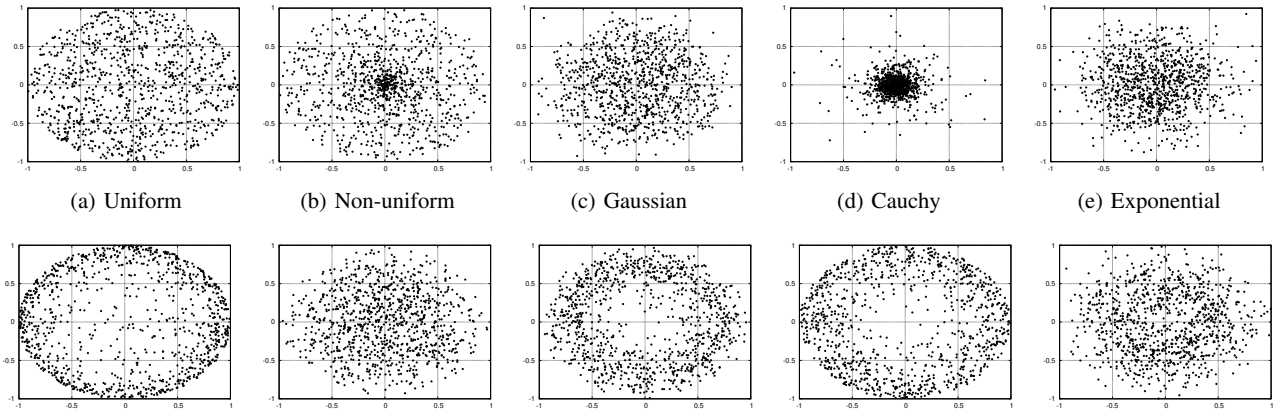
An effective search technique must strike a balance between exploration, by finding new and unexplored areas in the search space, and exploitation, by improving upon known and promising solutions. The balance between exploration and exploitation is even more important when considering dynamic environments. In the context of the quantum position update mechanism, this translates to striking a balance between generating positions close to the global best particle (exploitation) while also maintaining greater diversity by generating positions further from the global best (exploration). To examine the effects of a wide variety of such trade-offs, the following probability distributions were used to generate positions (calculated as an offset from $\hat{y}(t)$) for quantum particles.

- **Uniform**
- **Non-uniform** (linearly decreasing probability)
- **Gaussian** ($\mu = 0, \sigma = 0.38822$)
- **Cauchy** (location $= 0$, scale $= 0.01570$)
- **Exponential** (rate $= 4.60517$)
- **Beta** ($\alpha = 4.60517, \beta = 1$)
- **Triangular-0** (min $= 0$, max $= 1$, mode $= 0$)
- **Triangular-0.5** (min $= 0$, max $= 1$, mode $= 0.5$)
- **Triangular-1** (min $= 0$, max $= 1$, mode $= 1$)
- **Weibull** (shape $= 1.5$, scale $= 0.36127$)

For distributions supported on unbounded intervals, namely the Gaussian, Cauchy, exponential, and Weibull distributions, the respective parameters were selected such that 99% of samples drawn from the distribution have an absolute value less than 1. Note that when a distribution produces a sample with an absolute value greater than 1, this corresponds to generating a position which is outside of the quantum cloud. For the Beta distribution, the parameters were chosen to provide an approximate inverse for the exponential distribution. The parameters for the triangular distribution were selected to maximize exploration (mode $= 1$), exploitation (mode $= 0$), and an equal balance thereof (mode $= 0.5$), respectively. Figure 2 shows a sampling of positions around the origin, in two dimensions, using each probability distribution and a radius of 1.

IV. Experimental Procedure

To compare the performance of QPSO with the probability distributions, 24 benchmark environments were instantiated.



(a) Uniform (b) Non-uniform (c) Gaussian (d) Cauchy (e) Exponential

(f) Beta (g) Triangular-0 (h) Triangular-0.5 (i) Triangular-1 (j) Weibull

Fig. 2: 1000 sampled positions using various probability distributions and a radius of 1.

TABLE I: Environmental Control Parameters

Parameter	Environment Type			
	Quasi-Static	Progressive	Abrupt	Chaotic
Peaks	$1, 5$	$1, 5$	$1, 5$	$1, 5$
Peak Heights	$[30, 70]$	$[30, 70]$	$[30, 70]$	$[30, 70]$
Peak Widths	$[1, 12]$	$[1, 12]$	$[1, 12]$	$[1, 12]$
Height Severity	1	1	10	10
Width Severity	0.05	0.05	0.05	0.05
Change Severity (s)	1	1	50	50
λ	0	0	0	0
Change Freq. (Iterations)	50	10	50	10

TABLE II: Algorithmic Parameters

Parameter	Value
Particles	30
Quantum Proportion	50% (15 particles)
r_{cloud}	$\{0.5, 25\}$
ω	0.729844
$c1, c2$	1.496180
Topology	Von Neumann
Iteration Strategy	Synchronous

The moving peaks benchmark (MPB) [13] was used to generate both single- and five-peak dynamic environments, using parameters listed in Table I, to examine uni- and multi-modal landscapes, respectively. The selection of these parameters corresponds to the four dynamic environment types proposed by Duhain and Engelbrecht [7]. Furthermore, all environments are Type III, random, and homogeneous using the classification schemes of Eberhart *et al.* [11], [12], Angeline [8], and Weicker [10]. Experiments were performed using 10-, 30-, and 50-dimensional environments. Each QPSO strategy was executed 50 times in every environment for 1000 iterations using the parameters listed in Table II. The two examined values for r_{cloud}, namely 0.5 and 25, adhere to the recommendation of $r_{cloud} \approx s/2$ [4], [6] and represent exploitative and explorative behaviours, respectively. To remove the influence of change detection mechanisms, particles re-evaluated their personal best positions at each iteration. Finally, to prevent invalid attractors, a particle's personal best position was only updated if the new position was within the search space.

A. Performance Measures

When examining performance in dynamic environments, several aspects are important to measure, namely *accuracy* (the proximity to the optimum), *stability* (the reduction in solution quality when an environmental change occurs), *reactivity* (the ability of an optimizer to recover from an environmental change), and *exploitation capacity* (the ability to find quality solutions between environmental changes) [10], [15]. The five performance measures used in this study, described below, were selected to address each of these aspects of performance.

The *error* of an optimizer is the absolute value of the difference between the global best fitness and the optimal fitness. The error is a direct measure of an optimizer's accuracy.

The *collective mean error* (CME) is the mean error of the best solution over an entire simulation. The CME incorporates the accuracy, stability, reactivity, and (to a lesser degree) the exploitation capacity of an optimizer. An improvement in any of aforementioned aspects leads to an improvement in the CME value [15].

In a similar fashion to the CME, the *offline error* measures the mean error of the best solution across all iterations since the last environmental change occurred. The offline error incorporates the accuracy, reactivity, and exploitation capacity of an optimizer. However, the offline error requires knowledge of when an environmental change occurred, thus is it more well-suited to synthetic benchmarks compared to the CME.

The *average best error before change* (ABEBC) measures the average error of the best solution at each iteration just before an environmental change occurred. Note that if the environment changes every iteration, the ABEBC is equivalent to the CME. The ABEBC strongly measures the exploitation capacity of an optimizer.

Similar to the ABEBC, the *average best error after change* (ABEAC) measures the average error of the best solution at each iteration just after an environmental change occurred. The ABEAC strongly measures the stability of an optimizer.

B. Statistical Analysis

Statistical analysis of results was performed using the normalized wins and losses approach [16]. For each performance measure, a Kruskal-Wallis test was performed using the average performance measure values at each iteration just before an environmental change occurred. If the Kruskal-Wallis test indicated that a difference existed among the strategies, pairwise Mann-Whitney U tests were performed to identify the individual differences. When the Mann-Whitney U test indicated that a difference existed among two strategies, the average performance measure values at each iteration just before a change occurred were used to assign wins and losses; the better performing algorithm was awarded a win, while the inferior algorithm was awarded a loss. To prevent skewed results, the wins and losses are normalized using the number of environmental changes. Finally, the strategies are ranked based on the difference between the number of wins and losses.

V. Results and Discussion

Table III presents the results aggregated over all examined environments and dimensionalities. The triangular-0.5 distribution with a quantum radius of 0.5 resulted in the lowest overall error, while the remainder of measures (namely CME, offline error, ABEBC, and ABEAC) were all minimized most effectively when the non-uniform distribution with quantum radius of 0.5 was employed. The worst overall performance was noted when the Gaussian distribution with a radius of 25 was used. A further noteworthy observation was regarding the commonly used uniform distribution which, with respect to raw error, ranked 14th and 13th overall for $r_{cloud} = 0.5$ and $r_{cloud} = 25$, respectively. This indicates that the use of the typical uniform distribution within QPSO is by no means the most effective for the examined quantum radii. Additionally, the lower average ranks observed when $r_{cloud} = 0.5$ was used indicated that the smaller quantum radius was generally preferable to the larger radius. This suggests that, unsurprisingly, exploitative behaviour results in a lower raw error (and thus accuracy) than explorative behaviour, in general.

A. Performance by Environment Type

To identify the effect of each probability distribution on the performance of QPSO relative to the environment type, results were grouped by the four environmental classes described in Section II-A. The results are presented in Tables IV to VII.

Quasi-Static Environments: The results for quasi-static environments are presented in Table IV. For quasi-static environments, the lowest error was observed when the Cauchy distribution, with radius of 0.5, was used. The remainder of measures were most effectively minimized when the uniform distribution and $r_{cloud} = 0.5$ was used. However, the uniform distribution with a radius of 25 resulted in the worst raw error. For the remainder of performance measures, the Gaussian distribution with a radius of 25 performed the worst.

Progressive Environments: Table V presents the results for progressive environments. For all measures, the best performance was noted when a quantum radius of 25 was used – the best error was observed when the non-uniform distribution was employed while the remainder of measures were most effectively minimized through the use of the triangular-1 distribution. However, the worst performance was also observed when a radius of 25 was used, namely the Weibull distribution when error was examined and the exponential distribution for the remainder of measures.

Abrupt Environments: The results for abrupt environments are presented in Table VI. For all performance measures, the non-uniform distribution with a radius of 25 resulted in the best performance. The worst overall performance for all measures was observed when a Gaussian distribution, with radius 25, was employed.

Chaotic Environments: The results for chaotic environments are presented in Table VII. These results indicated that the best performance, with respect to all performance measures, occurred when the triangular-0.5 distribution was used with a quantum radius of 0.5. For all performance measures, the worst performance was demonstrated when the uniform distribution with a radius of 0.5 was employed.

Radius Size and Diversity: The average ranks relative to the radius size in Tables IV to VII show that for each of the quasi-static, progressive, and abrupt environments, the distributions which were coupled with the smaller quantum radius ($r_{cloud} = 0.5$) generally outperformed their counterparts with the larger quantum radius ($r_{cloud} = 25$). When chaotic environments were considered, the larger radius of 25 generally lead to better performance. These results indicated that when environmental changes were either relatively small in magnitude or relatively low in frequency, exploitative behaviour was favoured as opposed to explorative behaviour. However, when changes were both frequent and relatively large (i.e., chaotic environments), optimizers had little time to exploit the global best and, therefore, demonstrated superior performance when they could most effectively mitigate the drastic change in environment through heightened exploration.

To further support the observations regarding the radius size, Figure 3 shows the diversity profiles of the two best and two worst ranked distributions, with respect to error, for each environment type. Diversity is measured as the average distance from the swarm centre [17] and is reported on 30D, single-peak environments. Note that for quasi-static, progressive, and abrupt environments, the two distributions which performed best maintained a lower level of diversity than the distributions which lead to the worst performance. For chaotic environments, the distribution which showed the highest level of diversity (of those plotted) resulted in the second best performance on chaotic environments.

B. Performance in Single- vs Five-Peak Environments

Lastly, the results were grouped based on the modality of the environment to determine whether the number of peaks had a significant impact on performance. The results are shown in Tables VIII and IX, respectively.

Single-Peak Environments: When the results for single-peak environments were considered, as shown in Table VIII, it was noted that for four of the performance measures (namely error, CME, offline error, and ABEBC), the non-uniform distibution with radius of 0.5 resulted in the best performance. The ABEAC measure was optimized most effectively by the

TABLE III: Overall Difference and Rank for All Environments

r_{cloud}	Distribution	Error		CME		Offline Error		ABEBC		ABEAC	
		Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank
0.5	Beta	12.82	8	37.38	5	120.94	4	70.52	5	105.26	5
	Cauchy	19.86	4	20.40	10	28.36	11	20.48	12	21.12	12
	Exponential	16.86	6	46.82	3	119.26	5	77.34	4	109.80	4
	Gaussian	19.44	5	26.20	9	81.14	8	43.02	8	62.28	8
	Non-uniform	34.28	2	85.26	**1**	213.46	**1**	141.26	**1**	196.00	**1**
	Triangular-0.5	44.58	**1**	33.52	7	74.88	9	30.22	10	29.50	11
	Triangular-0	23.50	3	40.42	4	112.44	6	63.26	7	89.76	7
	Triangular-1	5.04	10	36.22	6	110.86	7	68.14	6	103.96	6
	Uniform	-12.48	14	32.00	8	126.60	3	84.96	3	138.70	3
	Weibull	8.46	9	-5.90	14	9.54	13	-3.84	13	0.04	13
	Average		6.2		6.7		6.7		6.9		7.0
25	Beta	-4.42	11	-5.26	13	-24.84	14	-13.26	14	-23.90	14
	Cauchy	-7.34	12	12.94	12	25.42	12	22.50	11	29.82	10
	Exponential	-19.30	17	-96.72	19	-283.22	19	-180.84	19	-266.20	19
	Gaussian	-42.68	20	-126.30	20	-348.14	20	-217.36	20	-308.72	20
	Non-uniform	14.16	7	58.78	2	159.42	2	105.26	2	150.84	2
	Triangular-0.5	-33.88	18	-61.30	18	-154.68	17	-94.18	17	-129.00	17
	Triangular-0	-14.74	15	-58.88	17	-168.40	18	-107.32	18	-156.70	18
	Triangular-1	-36.86	19	-41.16	15	-104.24	15	-55.76	15	-73.42	15
	Uniform	-10.98	13	14.48	11	38.60	10	31.46	9	46.42	9
	Weibull	-16.32	16	-48.90	16	-137.40	16	-85.86	16	-125.56	16
	Average		14.8		14.3		14.3		14.1		14.0

TABLE IV: Overall Difference and Rank for Quasi-Static Environments

r_{cloud}	Distribution	Error		CME		Offline Error		ABEBC		ABEAC	
		Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank
0.5	Beta	15.60	9	36.10	6	102.90	4	60.00	4	83.90	4
	Cauchy	29.10	**1**	36.90	5	69.10	8	43.60	7	50.90	8
	Exponential	25.40	2	32.70	7	66.80	9	39.30	9	47.30	9
	Gaussian	19.70	5	52.20	2	144.20	2	88.20	2	124.20	2
	Non-uniform	25.10	3	46.10	3	106.20	3	71.90	3	96.60	3
	Triangular-0.5	16.10	8	8.30	10	22.30	10	5.10	10	3.00	10
	Triangular-0	14.20	10	27.80	9	70.20	7	42.20	8	56.70	7
	Triangular-1	18.50	6	41.20	4	96.90	5	57.90	5	75.20	5
	Uniform	18.30	7	78.10	**1**	228.80	**1**	143.10	**1**	208.10	**1**
	Weibull	20.60	4	30.30	8	92.20	6	50.40	6	70.50	6
	Average		5.5		5.5		5.5		5.5		5.5
25	Beta	-21.40	16	-37.20	15	-90.30	15	-54.00	15	-71.40	15
	Cauchy	-20.80	15	-10.80	11	-21.70	11	-7.30	11	-3.80	11
	Exponential	-19.70	12	-38.40	16	-99.30	16	-59.20	16	-80.00	16
	Gaussian	-21.50	17	-74.80	20	-209.70	20	-132.00	20	-189.30	20
	Non-uniform	-3.30	11	-11.60	12	-34.90	12	-18.80	12	-26.50	12
	Triangular-0.5	-24.50	19	-37.00	14	-81.30	14	-50.20	14	-63.50	14
	Triangular-0	-20.10	13	-33.10	13	-77.00	13	-46.10	13	-59.10	13
	Triangular-1	-24.50	18	-58.20	19	-156.40	19	-97.10	19	-136.50	19
	Uniform	-26.50	20	-44.50	18	-113.90	17	-66.90	17	-89.30	17
	Weibull	-20.30	14	-44.10	17	-115.10	18	-70.10	18	-97.00	18
	Average		15.5		15.5		15.5		15.5		15.5

TABLE V: Overall Difference and Rank for Progressive Environments

r_{cloud}	Distribution	Error		CME		Offline Error		ABEBC		ABEAC	
		Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank
0.5	Beta	1.12	10	4.00	10	22.34	10	12.08	10	21.76	10
	Cauchy	1.92	7	9.82	8	27.18	9	18.36	9	26.88	9
	Exponential	1.68	8	15.92	7	45.84	7	30.52	7	45.66	7
	Gaussian	-1.92	14	-25.04	17	-58.62	16	-41.60	16	-55.72	16
	Non-uniform	2.30	6	25.80	5	74.06	5	49.52	5	73.08	5
	Triangular-0.5	2.38	5	-23.06	16	-66.76	17	-45.34	17	-66.14	17
	Triangular-0	-1.52	13	-4.44	12	1.98	11	-2.24	12	3.52	11
	Triangular-1	3.20	4	27.88	4	94.88	2	59.30	2	93.02	2
	Uniform	3.52	2	18.76	6	58.28	6	37.28	6	56.18	6
	Weibull	3.52	2	8.44	9	35.60	8	19.38	8	32.14	8
	Average		7.1		9.4		9.1		9.2		9.1
25	Beta	-4.52	19	-10.68	15	-40.70	15	-22.90	15	-37.16	15
	Cauchy	-3.10	15	-10.44	14	-38.86	14	-22.00	14	-35.80	14
	Exponential	-3.64	18	-70.82	20	-215.58	20	-141.70	20	-213.82	20
	Gaussian	1.18	9	-6.46	13	-25.70	13	-17.64	13	-29.02	13
	Non-uniform	4.94	**1**	30.74	2	87.90	3	57.20	3	83.34	3
	Triangular-0.5	-0.28	12	1.96	11	-3.90	12	-0.56	11	-4.92	12
	Triangular-0	-3.44	17	-29.06	19	-92.84	19	-58.98	19	-89.84	19
	Triangular-1	0.72	11	36.60	**1**	98.98	**1**	67.38	**1**	96.80	**1**
	Uniform	-3.26	16	28.22	3	79.36	4	55.60	4	80.96	4
	Weibull	-4.80	20	-28.14	18	-83.44	18	-53.66	18	-80.92	18
	Average		13.8		11.6		11.9		11.8		11.9

TABLE VI: Overall Difference and Rank for Abrupt Environments

r_{cloud}	Distribution	Error		CME		Offline Error		ABEBC		ABEAC	
		Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank
0.5	Beta	0.90	12	9.10	8	26.40	7	17.30	7	25.50	6
	Cauchy	-15.00	18	-33.40	19	-85.30	19	-51.80	19	-70.20	19
	Exponential	-1.20	14	11.60	6	37.80	5	25.30	5	39.00	4
	Gaussian	10.30	2	18.20	4	44.40	4	26.10	4	34.00	5
	Non-uniform	-9.80	17	-20.50	17	-51.70	17	-31.20	17	-41.90	16
	Triangular-0.5	1.80	10	-1.00	14	-4.20	13	-3.80	15	-6.60	14
	Triangular-0	3.30	9	0.40	11	-2.20	12	-2.50	13	-5.40	13
	Triangular-1	7.20	4	23.10	2	63.10	2	39.00	2	55.90	2
	Uniform	1.30	11	-0.20	13	-2.10	11	-1.70	11	-2.80	12
	Weibull	6.90	5	10.60	7	24.90	8	14.30	9	18.00	9
	Average		10.2		10.1		9.8		10.2		10.0
25	Beta	5.50	6	5.00	10	9.30	10	4.40	10	3.80	10
	Cauchy	-3.50	16	-2.90	15	-5.20	15	-2.30	12	-1.70	11
	Exponential	-16.30	19	-29.90	18	-75.20	18	-44.40	18	-58.90	18
	Gaussian	-18.50	20	-42.20	20	-108.10	20	-65.90	20	-89.60	20
	Non-uniform	10.90	**1**	26.70	**1**	69.20	**1**	42.60	**1**	58.40	**1**
	Triangular-0.5	0.90	13	8.40	9	24.50	9	15.90	8	23.40	7
	Triangular-0	-2.10	15	-15.70	16	-44.60	16	-29.30	16	-42.90	17
	Triangular-1	4.50	8	0.40	12	-4.50	14	-3.70	14	-9.00	15
	Uniform	7.70	3	12.90	5	30.60	6	18.10	6	23.30	8
	Weibull	5.20	7	19.40	3	52.90	3	33.60	3	47.70	3
	Average		10.8		10.9		11.2		10.8		11.0

TABLE VII: Overall Difference and Rank for Chaotic Environments

r_{cloud}	Distribution	Error		CME		Offline Error		ABEBC		ABEAC	
		Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank	Difference	Rank
0.5	Beta	-4.80	13	-11.82	13	-30.70	13	-18.86	14	-25.90	15
	Cauchy	3.84	9	7.08	10	17.38	10	10.32	10	13.54	10
	Exponential	-9.02	15	-13.40	14	-31.18	14	-17.78	13	-22.16	13
	Gaussian	-8.64	14	-19.16	15	-48.84	16	-29.68	16	-40.20	16
	Non-uniform	16.68	4	33.86	5	84.90	5	51.04	5	68.22	5
	Triangular-0.5	24.30	**1**	49.28	**1**	123.54	**1**	74.26	**1**	99.24	**1**
	Triangular-0	7.52	8	16.66	8	42.46	8	25.80	7	34.94	8
	Triangular-1	-23.86	19	-55.96	19	-144.02	19	-88.06	19	-120.16	18
	Uniform	-35.60	20	-64.66	20	-158.38	20	-93.72	20	-122.78	20
	Weibull	-22.56	18	-55.24	18	-143.16	18	-87.92	18	-120.60	19
	Average		12.1		12.3		12.4		12.3		12.5
25	Beta	16.00	5	37.62	3	96.86	3	59.24	3	80.86	3
	Cauchy	20.06	3	37.08	4	91.18	4	54.10	4	71.12	4
	Exponential	20.34	2	42.40	2	106.86	2	64.46	2	86.52	2
	Gaussian	-3.86	12	-2.84	12	-4.64	12	-1.82	12	-0.80	12
	Non-uniform	1.62	11	12.94	9	37.22	9	24.26	9	35.60	6
	Triangular-0.5	-10.00	16	-34.66	17	-93.98	17	-59.32	17	-83.98	17
	Triangular-0	10.90	7	18.98	6	46.04	6	27.06	6	35.14	7
	Triangular-1	-17.58	17	-19.96	16	-42.32	15	-22.34	15	-24.72	14
	Uniform	11.08	6	17.86	7	42.54	7	24.66	8	31.46	9
	Weibull	3.58	10	3.94	11	8.24	11	4.30	11	4.66	11
	Average		8.9		8.7		8.6		8.7		8.5

triangular-1 distribution with radius of 25. The worst performance for all measures was demonstrated by the Gaussian distribution with radius of 25.

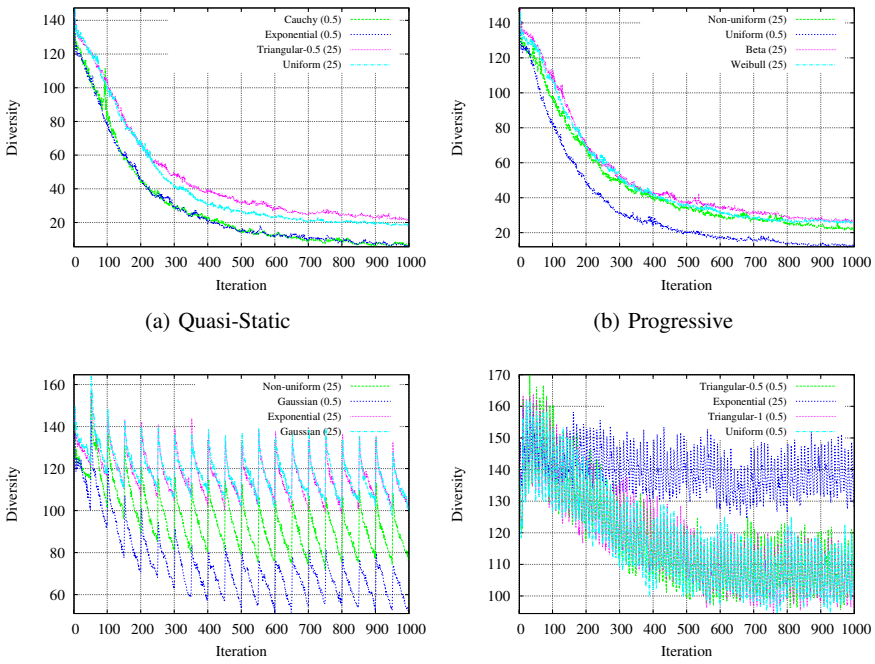
The average ranks in Table VIII depict that, in general, the radius of 0.5 was preferable to the radius of 25. Given that the task in single-peak environments is to track and optimize a single peak, it is not unexpected to observe that exploitation is generally preferred. Once the lone peak has been found, the exploratory phase has essentially ended, and exploitation becomes more important. Therefore, a larger capacity for exploitation exists in uni-modal environments than multi-modal environments.

Five-Peak Environments: The results for five-peak environments are shown in Table IX. The raw error was optimized most effectively by the triangular-0.5 distribution with a radius of 0.5 while the non-uniform distribution, with radius 25, demonstrated the best performance with respect to the remainder of measures. The triangular-1 distribution with radius 25 resulted in the worst overall values for error, CME, offline error, and ABEBC. The worst ABEAC measure was demonstrated by the exponential distribution with radius 25.

As with single peak-environments, distributions coupled with a radius of 0.5 resulted in lower average ranks than those coupled with a radius of 25. This indicates that even when multi-modal environments are examined, excessive exploration can be detrimental to performance.

C. Summary

When aggregated across all environments, the triangular-0.5 distribution with a radius of 0.5 lead to the best error, and thus accuracy, while a non-uniform distribution with radius of 0.5 demonstrated the best performance with respect to all other measures. It was observed that for all but chaotic environments, the smaller radius of 0.5 lead to superior performance, in general. For abrupt environments, the smaller radius directly contradicted the suggested value of $r_{cloud} = s/2$ [4], [6].



(a) Quasi-Static

(b) Progressive

(c) Abrupt

(d) Chaotic

Fig. 3: Average swarm diversity profiles for the two distributions with the best error and the two distributions with the worst error, in that order, on 30D, single-peak environments. Values in parentheses denote the size of r_{cloud}.

While further evidence is need to conclusively state the fact, these results indicated that exploitation is generally preferred to exploration in QPSO, contrary to previous assumptions [6].

A noteworthy observation was that in no type of environment did the commonly used uniform distribution (with either radius size) lead to lowest overall error. However, the uniform distribution with a radius of 0.5 did result in the best CME, offline error, ABEBC, and ABEAC measures on quasi-static environments. This observation indicates that, in general, applications which make use of the standard QPSO algorithm can be improved upon by using a more effective probability distribution when generating the quantum positions.

VI. Conclusion

This paper presented an empirical study on the performance implications of different probability distributions being used to generate the positions of quantum particles in the quantum particle swarm optimization (QPSO) algorithm. Ten probability distributions were examined using two quantum radius sizes on a wide variety of dynamic environments with various temporal and spatial severities, dimensionalities, and peaks.

The results indicated that the most effective probability distribution was dependent upon the type of dynamism. However, with the exception of chaotic environments, a smaller radius, and thereby a more compact quantum cloud, lead to superior performance, in general. Given that the smaller radius, with heavier focus on exploitation, lead to better performance, this provided evidence to refute previous assumptions that the task of quantum particles is purely exploration. A final noteworthy conclusion was that the uniform distribution performed relatively poorly overall, especially when considering solution accuracy. Therefore, the use of alternative probability distributions for updating quantum positions may lead to performance improvements in many applications which employ QPSO.

The primary objective of this study was to provide empirical evidence that examining alternative probability distributions for quantum particles is warranted. A future study will aim to generalize the results and give direction as to which probability distributions are better suited to specific environment types. Additionally, further work will examine the performance of the probability distributions relative to the QPSO parameters, namely the proportion of quantum particles and the quantum radius. It is not unreasonable to expect that given a different number of quantum particles, the optimal distribution for them would differ. Similarly, having a different sized hypersphere to generate positions may also influence the distribution which is most effective. Another study will consider multi-peak environments in a more focused fashion, i.e. using multi-swarm QPSO variants.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the 1995 IEEE International Joint Conference on Neural Networks*, vol. IV, 1995, pp. 1942–1948.

[2] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. Springer, 2010, pp. 760–766.

[3] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization," *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, 2007.

TABLE VIII: Overall Difference and Rank for Single-Peak Environments

r_{cloud}	Distribution	Error Difference	Error Rank	CME Difference	CME Rank	Offline Error Difference	Offline Error Rank	ABEBC Difference	ABEBC Rank	ABEAC Difference	ABEAC Rank
0.5	Beta	9.28	5	24.54	4	69.14	4	40.68	5	57.72	5
	Cauchy	8.28	6	0.44	13	-17.22	13	-8.16	13	-16.62	14
	Exponential	7.56	7	9.76	9	31.52	8	16.48	10	25.18	9
	Gaussian	20.92	2	36.96	2	105.54	2	59.44	3	82.32	3
	Non-uniform	28.88	**1**	53.06	**1**	122.76	**1**	77.84	**1**	101.58	2
	Triangular-0.5	17.38	3	5.80	10	16.64	12	-0.08	12	-3.32	12
	Triangular-0	6.26	8	12.24	8	33.28	7	19.18	8	26.54	8
	Triangular-1	-3.14	13	20.76	5	66.48	5	44.02	4	67.94	4
	Uniform	-21.90	19	1.52	12	31.42	9	27.48	7	53.94	6
	Weibull	-2.04	12	3.68	11	17.60	11	11.08	11	18.70	11
	Average		7.6		7.5		7.2		7.4		7.4
25	Beta	-21.12	18	-40.54	19	-111.92	19	-65.54	19	-92.00	19
	Cauchy	4.12	9	13.04	7	29.78	10	18.40	9	24.12	10
	Exponential	-9.20	16	-33.12	17	-96.64	17	-60.62	17	-88.56	17
	Gaussian	-30.96	20	-79.46	20	-214.68	20	-132.36	20	-185.32	20
	Non-uniform	13.38	4	18.66	6	47.12	6	27.68	6	37.18	7
	Triangular-0.5	-15.70	17	-38.56	18	-100.46	18	-63.56	18	-89.62	18
	Triangular-0	2.72	10	-5.82	14	-22.68	14	-15.86	15	-26.14	15
	Triangular-1	-1.06	11	35.94	3	104.02	3	71.04	2	104.10	**1**
	Uniform	-8.94	15	-9.98	15	-25.62	15	-12.14	14	-15.00	13
	Weibull	-4.72	14	-28.92	16	-86.08	16	-55.00	16	-82.74	16
	Average		13.4		13.5		13.8		13.6		13.6

TABLE IX: Overall Difference and Rank for Five-Peak Environments

r_{cloud}	Distribution	Error Difference	Error Rank	CME Difference	CME Rank	Offline Error Difference	Offline Error Rank	ABEBC Difference	ABEBC Rank	ABEAC Difference	ABEAC Rank
0.5	Beta	3.54	10	12.84	11	51.80	9	29.84	9	47.54	8
	Cauchy	11.58	4	19.96	9	45.58	10	28.64	10	37.74	9
	Exponential	9.30	7	37.06	2	87.74	4	60.86	3	84.62	4
	Gaussian	-1.48	12	-10.76	14	-24.40	14	-16.42	14	-20.04	14
	Non-uniform	5.40	9	32.20	4	90.70	3	63.42	2	94.42	2
	Triangular-0.5	27.20	**1**	27.72	7	58.24	8	30.30	8	32.82	11
	Triangular-0	17.24	2	28.18	6	79.16	6	44.08	6	63.22	6
	Triangular-1	8.18	8	15.46	10	44.38	11	24.12	11	36.02	10
	Uniform	9.42	6	30.48	5	95.18	2	57.48	4	84.76	3
	Weibull	10.50	5	-9.58	13	-8.06	13	-14.92	13	-18.66	13
	Average		6.4		8.1		8.0		8.0		8.0
25	Beta	16.70	3	35.28	3	87.08	5	52.28	5	68.10	5
	Cauchy	-11.46	15	-0.10	12	-4.36	12	4.10	12	5.70	12
	Exponential	-10.10	14	-63.60	19	-186.58	19	-120.22	19	-177.64	20
	Gaussian	-11.72	17	-46.84	17	-133.46	17	-85.00	17	-123.40	17
	Non-uniform	0.78	11	40.12	**1**	112.30	**1**	77.58	**1**	113.66	**1**
	Triangular-0.5	-18.18	19	-22.74	16	-54.22	16	-30.62	15	-39.38	15
	Triangular-0	-17.46	18	-53.06	18	-145.72	18	-91.46	18	-130.56	18
	Triangular-1	-35.80	20	-77.10	20	-208.26	20	-126.80	20	-177.52	19
	Uniform	-2.04	13	24.46	8	64.22	7	43.60	7	61.42	7
	Weibull	-11.60	16	-19.98	15	-51.32	15	-30.86	16	-42.82	16
	Average		14.6		12.9		13.0		13.0		13.0

[4] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 459–472, 2006.

[5] T. M. Blackwell, P. J. Bentley *et al.*, "Dynamic search with charged swarms," in *Proceedings of the 2002 Genetic and Evolutionary Computation Conference*, 2002, pp. 19–26.

[6] T. Blackwell, J. Branke, and X. Li, "Particle swarms for dynamic optimization problems," in *Swarm Intelligence*, ser. Natural Computing Series. Springer, 2008, pp. 193–217.

[7] J. Duhain and A. Engelbrecht, "Towards a more complete classification system for dynamically changing environments," in *Proceedings of the 2012 IEEE Congress on Evolutionary Computation*, 2012, pp. 1–8.

[8] P. J. Angeline, "Tracking extrema in dynamic environments," in *Proceedings of Evolutionary Programming VI*, ser. Lecture Notes in Computer Science, vol. 1213. Springer, 1997, pp. 335–345.

[9] K. De Jong, "Evolving in a changing world," in *Foundations of Intelligent Systems*, ser. Lecture Notes in Computer Science. Springer, 1999, vol. 1609, pp. 512–519.

[10] K. Weicker, "Performance measures for dynamic environments," in *Proceedings of the 7th International Conference on Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science. Springer, 2002, vol. 2439, pp. 64–73.

[11] R. C. Eberhart and Y. Shi, "Tracking and optimizing dynamic systems with particle swarms," in *Proceedings of the 2001 IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2001, pp. 94–100.

[12] X. Hu and R. Eberhart, "Tracking dynamic systems with pso: wheres the cheese," in *Proceedings of the Workshop on Particle Swarm Optimization*, 2001, pp. 80–83.

[13] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in *Proceedings of the 1999 IEEE Congress on Evolutionary Computation*, vol. 3, 1999.

[14] J. Branke, T. Kaußler, C. Smidt, and H. Schmeck, "A multi-population approach to dynamic optimization problems," in *Evolutionary Design and Manufacture*. Springer, 2000, pp. 299–307.

[15] J. Duhain, "Particle swarm optimisation in dynamically changing environments – an emprical study," Master's thesis, U. Pretoria, 2011.

[16] M. Helbig and A. Engelbrecht, "Analysing the performance of dynamic multi-objective optimisation algorithms," in *Proceedings of the 2013 IEEE Congress on Evolutionary Computation*, 2013, pp. 1531–1539.

[17] O. Olorunda and A. Engelbrecht, "Measuring exploration/exploitation in particle swarms using swarm diversity," in *Proceedings of the 2008 IEEE Congress on Evolutionary Computation*, 2008, pp. 1128–1134.