Geometric Particle Swarm Optimisation on Binary and Real Spaces: from Theory to Practice

Cecilia Di Chio
Department of Computer Science
University of Essex, UK
cdichi@essex.ac.uk

Alberto Moraglio
Department of Computer Science
University of Essex, UK
amoragn@essex.ac.uk

Riccardo Poli
Department of Computer Science
University of Essex, UK
rpoli@essex.ac.uk

ABSTRACT

Geometric particle swarm optimization (GPSO) is a recently introduced formal generalization of traditional particle swarm optimization (PSO) that applies naturally to both continuous and combinatorial spaces. In previous work we have developed the theory behind it. The aim of this paper is to demonstrate the applicability of GPSO in practice. We demonstrate this for the cases of Euclidean, Manhattan and Hamming spaces and report extensive experimental results.

Categories and Subject Descriptors

F.2 [**Theory of Computation**]: Analysis of Algorithms and Problem Complexity

General Terms

Theory

Keywords

Particle Swarm Optimisation, Metric Space, Geometric Crossover

1. INTRODUCTION

Particle swarm optimisation [5] has traditionally been applied to continuous search spaces. Although a version of PSO for binary search spaces has been defined [4], attempts to extend PSO to richer spaces, such as, for example, combinatorial spaces, have had no real success [2].

There are two ways of extending PSO to richer spaces: a) adapting the PSO for each new solution representation, or b) making use of a rigorous mathematical generalisation to a general class of spaces of the notion (and motion) of particles. This second approach has the advantage that a PSO can be derived in a principled way for any search space belonging to the given class. In recent work [13] we have pursued this approach. We have shown *formally* how a general form of PSO (without the momentum term), the Geometric PSO (GPSO), can be obtained by using theoretical tools developed for a different form of search algorithms, namely evolutionary algorithms using geometric crossover and geometric mutation. These are representation-independent operators that generalise many pre-existing search operators for the major representations, such as binary strings [7], real vectors [7], permutations [8], syntactic trees [8] and sequences [10]. We have then formally derived GPSOs for Euclidean, Manhattan and Hamming spaces and discussed how to derive GPSOs for virtually any representation in a similar way. We did not, however, test these new PSOs.

In this paper we test GPSO theory experimentally: we implement the specific GPSO for Euclidean, Manhattan and Hamming spaces and report extensive experimental results obtaining very good performances.

For completeness, in the first part of the paper we extensively summarise the results of [13]. In particular: in section 2, we introduce the geometric framework; in section 3, we review the general GPSO algorithm for generic metric spaces; in section 4 we review the theory for the specific GPSOs for Euclidean, Manhattan and Hamming spaces; finally, in section 5, we discuss how to specialise the general GPSO automatically to virtually any solution representation using geometric crossover. Then, in section 6, we report new and extensive experimental results with the GPSOs for Euclidean, Manhattan and Hamming spaces and we compare them with a traditional PSO. Finally, in section 7, we present conclusions and future work.

2. GEOMETRIC FRAMEWORK

Geometric operators are defined in geometric terms using the notions of line segment and ball. These notions and the corresponding genetic operators are well-defined once a notion of distance in the search space is defined.

2.1 Geometric preliminaries

In the following we give necessary preliminary geometric definitions and extend those introduced in [7]. For more details on these definitions see [3].

The terms *distance* and *metric* denote any real valued function that conforms to the axioms of identity, symmetry and triangular inequality. A simple connected graph is naturally associated to a metric space via its *path metric*: the distance between two nodes in the graph is the length of a shortest path between the nodes. Distances arising from graphs via their path metric are called *graphic distances*.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
GECCO'07, July 7–11, 2007, London, England, United Kingdom.
Copyright 2007 ACM 978-1-59593-698-1/07/0007 ...$5.00.

Similarly, a weighted graph with positive weights is naturally associated to a metric space via a *weighted path metric*.

In a metric space (S, d) a *closed ball* is a set of the form $B(x;r) = \{y \in S | d(x,y) \leq r\}$ where $x \in S$ and r is a positive real number. A *line segment* is a set of the form $[x;y] = \{z \in S | d(x,z) + d(z,y) = d(x,y)\}$ where $x, y \in S$ are called extremes of the segment. Metric ball and metric segment generalise the familiar notions of ball and segment in the Euclidean space to any metric space through distance redefinition. In general, there may be more than one shortest path (*geodesic*) connecting the extremes of a metric segment; the metric segment is the union of all geodesics.

We assign a structure to the solution set by endowing it with a notion of distance d. $M = (S, d)$ is then a solution *space* and $L = (M, g)$ is the corresponding *fitness landscape*.

2.2 Geometric crossover

DEFINITION 1. *(Geometric crossover) A binary operator is a geometric crossover under the metric d if all offspring are in the segment between its parents.*

The definition is *representation-independent* and, therefore, crossover is well-defined for any representation. Being based on the notion of metric segment, *crossover is only function of the metric d* associated with the search space.

This class of operators is very broad. Blend and line crossovers, box recombinations, and discrete recombinations for real vectors are geometric crossovers [7]. For binary and multary strings, all homologous crossovers are geometric [7, 11]. For permutations, PMX, Cycle crossover, merge crossover and others are geometric crossovers [8]. For syntactic trees, the family of homologous crossovers are geometric [9]. Recombinations for several more complex representations are also geometric [10, 7, 8, 12].

2.3 Extension to multi-parent geometric crossover

To extend geometric crossover to the case of multiple parents we need the following definitions.

A family $\mathcal{X}$ of subsets of a set X is called *convexity on* X if: (C1) the empty set $\emptyset$ and the universal set X are in $\mathcal{X}$, (C2) if $\mathcal{D} \subseteq \mathcal{X}$ is non-empty, then $\bigcap \mathcal{D} \in \mathcal{X}$, and (C3) if $\mathcal{D} \subseteq \mathcal{X}$ is non-empty and totally ordered by inclusion, then $\bigcup \mathcal{D} \in \mathcal{X}$. The pair $(X, \mathcal{X})$ is called *convex structure*. The members of $\mathcal{X}$ are called *convex sets*. By the axiom (C1) a subset A of X of the convex structure is included in at least one convex set, namely X. From axiom (C2), A is included in a smallest convex set, the *convex hull* of A: $co(A) = \bigcap\{C | A \subseteq C \in \mathcal{X}\}$. The convex hull of a finite set is called a *polytope*. The axiom (C3) requires *domain finiteness* of the convex hull operator: a set C is convex iff it includes $co(F)$ for each finite subset F of C. The convex hull operator applied to set of cardinality two is called *segment operator*. Given a metric space $M = (X, d)$ the segment between a and b is the set $[a, b]_d = \{z \in X | d(x, z) + d(z, y) = d(x, y)\}$. The abstract *geodetic convexity* $\mathcal{C}$ on X induced by M is obtained as follow: a subset C of X is geodetically-convex provided $[x, y]_d \subseteq C$ for all x, y in C. If co denotes the convex hull operator of $\mathcal{C}$, then $\forall a, b \in X : [a, b]_d \subseteq co\{a, b\}$. The two operators need not to be equal: there are metric spaces in which metric segments are not all convex.

We can now provide the following extension:

DEFINITION 2. *(Multi-parent geometric crossover) In a multi-parent geometric crossover, given n parents $p_1, \ldots, p_n$ their offspring are contained in the metric convex hull of the parents $co(\{p_1, p_2, \ldots, p_n\})$ for some metric d.*

THEOREM 1. *(Decomposable three-parent recombination) Every multi-parent recombination $RX(p_1, p_2, p_3)$ that can be decomposed as a sequence of 2-parent geometric crossovers under the same metric GX and GX', so $RX(p_1, p_2, p_3) = GX(GX'(p_1, p_2), p_3)$, is a three-parent geometric crossover.*

(Proofs of this and other theorems can be found in [13].)

3. GEOMETRIC PSO

3.1 Basic, Canonical PSO Algorithm

Consider the canonical global topology PSO with inertia weight, where velocities are updated using the rule

$$\begin{aligned} v_i(t+1) &= \omega v_i(t) + \phi_1 R_1(x_{g_i}(t) - x_i(t)) \\ &+ \phi_2 R_2(x_{p_i}(t) - x_i(t)) \end{aligned} \quad (1)$$

The main feature that allows the motion of particles is the ability to perform linear combinations of points in the search space. To obtain a generalisation of PSO to generic search spaces, we can achieve this same ability by using multiple (geometric) crossover operations.

Consider a particle at position p that moves in the direction to a point o being in the next time step in position p'. We could interpret the motion of this particle as the result of the application of geometric crossover. That is, p and o can be seen as two parents and p' can be seen as the offspring. We can interpret the distance between parent p and offspring p' as the intensity of the velocity of the particle. Notice that the particle moves from p to p', so we see this process as result of a geometric crossover, we must imagine that parent p is replaced by the offspring p' in the next time step. Since all particles move at the same time, we must also imagine that all are selected for mating.

Similarly we can interpret the motion produced by the application of Equation (1) as the result of the application of a weighted multi-recombination involving the best position visited by the particle and the best position visited by its neighbours. Weights are the propensity of a particle towards memory, sociality, stability.

Naturally, particle motion based on geometric crossover leads to a form of search that cannot extend beyond the convex hull of the initial population. Mutation can be used to allow non-convex search.

We explain these ideas in detail in the following sections.

3.2 Geometric interpretation of linear combinations

If $v_1, ..., v_n$ are vectors and $a_1, ..., a_n$ are scalars, then the *linear combination* of those vectors with those scalars as coefficients is: $\sum_{i=1}^{n} a_i v_i$. A linear combination on n linearly independent vectors spans completely a n-dimensional or lower dimensional space but not a higher dimensional one.

An *affine combination* of vectors $x_1, ..., x_n$ is a linear combination $\sum_{i=1}^{n} \alpha_i \cdot x_i$ in which $\sum_{i=1}^{n} \alpha_i = 1$. When a vector represents a point in space, the affine combination of 2 points spans completely the line passing through them; the affine combination of 3 points spans the plane (a 2–D line) passing through them; increasing number of points spans higher dimensional "lines".

A *convex combination* is a linear combination of vectors where $\alpha_i \geq 0$ and $\sum_{i=1}^{n} \alpha_i = 1$. When vectors represent points in space, all possible convex combinations (given the base vectors) form the *convex hull*. For $n = 2$ a new point formed by the convex combination lies on a straight line between x_1 and x_2, while for $n = 3$ the convex hull is the triangle with vertices x_1, x_2 and x_3.

3.3 Simplified PSO with implicit velocity

THEOREM 2. *In a PSO with no momentum ($\omega = 0$) and where learning rates are such that $\phi_1 + \phi_2 < 1$, the future position of each particle x' is within the triangle formed by its current position x, its local best $\hat{x}$ and the swarm best $\hat{g}$. Furthermore, x' can be expressed without involving the particle's velocity as $x' = (1 - w_2 - w_3)x + w_2\hat{x} + w_3\hat{g}$.*

In the next section, we generalize this simplified form of PSO from real vectors to generic metric spaces. Mutation will be required to extend the search beyond the convex hull.

3.4 Convex combinations in metric spaces

Linear combinations are well-defined for vector spaces, algebraic structures endowed with scalar product and vectorial sum. A metric space is a set endowed with a notion of distance. The set underlying a metric space does not normally come with well-defined notions of scalar product and sum among its elements. So a linear combination of its elements is not defined. How can we then define a convex combination in a metric space? Vectors in a vector space can be easily understood as points in a metric space. However, what do the scalar weights in a convex combination mean in a metric space?

As seen in section 3.2, a convex combination is an algebraic description of a convex hull. However, even if the notion of convex combination is not defined for metric spaces, convexity in metric spaces is still well-defined through the notion of metric convex set that is a straightforward generalization of traditional convex set. Then, we can generalize the scalar weights of a convex combination making sense of them in terms of distance.

The weight of a point in a convex combination can be seen as a measure of relative linear attraction toward its corresponding point versus attractions toward the other points of the combination. The closer the weight to one, the stronger the attraction to its corresponding point. The resulting point of the convex combination can be seen as a weighted spatial average and it is the equilibrium point of all the attraction forces. The distance between the equilibrium point and a point of the convex combination is therefore a decreasing function of the level of attraction, weight, of the point: the stronger the attraction, the smaller its distance to the equilibrium point. This observation can be used to reinterpret the weights of a convex combination in a metric space as follows: $y = w_1x_1 + w_2x_2 + w_3x_3$ with w_1, w_2 and w_3 greater than zero and $w_1 + w_2 + w_3 = 1$ is generalized to $d(x_1, y) \sim 1/w_1$, $d(x_2, y) \sim 1/w_2$ and $d(x_3, y) \sim 1/w_3$.

This definition is formal and valid for all metric spaces but it is non-constructive. In contrast a convex combination, not only defines a convex hull, but it tells also how to reach all its points. So, how can we actually pick a point in the convex hull respecting the above distance requirements? Geometric crossover will help us with this.

The requirements for a convex combination in a metric space are:

1. Convex Weights: the weights respect the form of a convex combination: $w_1, w_2, w_3 > 0$ and $w_1 + w_2 + w_3 = 1$
2. Convexity: the convex combination operator combines x, $\hat{x}$ and $\hat{g}$ and returns a point in their metric convex hull, or simply triangle, under the metric of the space considered
3. Coherence between weights and distances: the distances to the equilibrium point are decreasing functions of their weights
4. Symmetry: the same value assigned to w_1, w_2 or w_3 has the same weight (so in a equilateral triangle, if the coefficients have all the same value, the distance to the equilibrium point are the same)

4. GEOMETRIC PSO FOR SPECIFIC SPACES

4.1 Euclidean space

GPSO for the Euclidean space is not an extension of the traditional PSO. We include it to show how the general notions introduced in the previous section materialize in a familiar context. The convex combination operator for the Euclidean space is the traditional convex combination that produces points in the traditional convex hull. Notice that when the definition of metric convex hull is specified for the case of the Euclidean distance we obtain the traditional convex hull.

In section 3.4, we have mentioned how to interpret the weights in a convex combination in terms of distances. The following result shows analytically how the weights of a convex combination affect the relative distances to the equilibrium point:

THEOREM 3. *In a convex combination, the distances to the equilibrium point are decreasing functions of the corresponding weights.*

The traditional convex combination in the Euclidean space respects the four requirements for a convex combination presented in section 3.4.

4.2 Manhattan space

Let us first define a multi-parent geometric recombination for the Manhattan space.

DEFINITION 3. *(Box recombination family) Given two parents a and b in $\mathbb{R}^n$, a box recombination operator returns offspring o such as $i = 1 \dots n : o_i \in [min(a_i, b_i), max(a_i, b_i)]$*

THEOREM 4. *(Geometricity of box recombination) Any box recombination is geometric crossover under Manhattan distance*

DEFINITION 4. *(Multi-parent Box recombination family) Given three parents a, b and c in $\mathbb{R}^n$, a box recombination operator returns offspring o such as $i = 1 \dots n : o_i \in [min(a_i, b_i, c_i), max(a_i, b_i, c_i)]$*

THEOREM 5. *(Geometricity of multi-parent box recombination) Any multi-parent box recombination is geometric crossover under Manhattan distance*

DEFINITION 5. *(Weighted multi-parent Box recombination) Given three parents a, b and c in $\mathbb{R}^n$ and weights w_a, w_b and w_c, a weighted box recombination operator returns offspring o such as $i = 1 \ldots n : o_i = w_{a_i} a_i + w_{b_i} b_i + w_{c_i} c_i$ where w_{a_i}, w_{b_i} and w_{c_i} are a convex combination of randomly perturbed weights with expected values w_a, w_b and w_c.*

The difference between box recombination and linear recombination (Euclidean space) is that in the latter the weights w_a, w_b and w_c are randomly perturbed only once and the same weights are used for all the dimensions, whereas the former one has a different randomly perturbed version of the weights for each dimension.

The weighted multi-parent box recombination belongs to the family of multi-parent box recombination because $i = 1 \ldots n : o_i = w_{a_i} a_i + w_{b_i} b_i + w_{c_i} \in [min(a_i, b_i, c_i), max(a_i, b_i, c_i)]$, hence it is geometric.

THEOREM 6. *(Coherence between weights and distances) In weighted multi-parent box recombination, the distances of the parents to the expected offspring are decreasing functions of the corresponding weights.*

So, the weighted multi-parent box recombination is a convex combination operator satisfying the four requirements of a metric convex combination for the Manhattan space (see section 3.4).

4.3 Hamming space

Let us first define a multi-parent recombination for binary strings that is a straightforward generalization of mask-based crossover with two parents.

DEFINITION 6. *(Multi-parent mask-based crossover family) Given three parents a, b and c in $0, 1^n$, generate randomly a crossover mask of length n with symbols from the alphabet a, b, c. Build the offspring o filling it in each position with the bit from the parent appearing in the crossover mask at the position.*

The weights w_a, w_b and w_c of the convex combination indicate for each position in the crossover mask the probability of having the symbols a, b or c.

THEOREM 7. *(Geometricity of multi-parent mask-based crossover) Any multi-parent mask-based crossover is geometric crossover under Hamming distance*

THEOREM 8. *(Coherence between weights and distances) In weighted multi-parent mask-based crossover, the distances of the parents to the expected offspring are decreasing functions of the corresponding weights.*

So, the weighted multi-parent mask-based crossover is a convex combination operator satisfying the four requirements of a metric convex combination for the Hamming space (section 3.4).

5. GEOMETRIC PSO FOR OTHER REPRESENTATIONS

Before introducing how to extend GPSO for other solution representations, we will discuss the relation between 3-parent geometric crossover and the symmetry requirement for a convex combination.

For each of the spaces in section 4, we have first considered, or defined, a three-parent recombination and then we proved that it is a three-parent geometric crossover by showing that it can be actually decomposed into two sequential applications of a geometric crossover for the specific space.

However, we could have skipped altogether the *explicit definition* of a three-parent recombination. In fact to obtain the three-parent recombination we could have used two sequential applications of a known two-parent geometric crossover for the specific space. This composition is indeed a three-parent recombination, it combines three parents, and it is decomposable by construction, hence it is a three-parent geometric crossover. This, indeed, would have been simpler than the route we took.

The reason we preferred to define explicitly a three-parent recombination is that the requirement of symmetry of the convex combination is true by construction: if the roles of any two parents are swapped exchanging in the three-parent recombination both positions and respective recombination weights, the resulting recombination operator is equivalent to the original operator.

The symmetry requirement becomes harder to enforce and prove for a three-parent geometric crossover obtained by two sequential applications of a two-parent geometric crossover (see [13]).

Let us now turn again to the extension of geometric PSO to other representations. We have seen that there are two alternative ways to produce a convex combination for a new representation: (i) explicitly define a symmetric three-parent recombination anew for the new representation and then prove its geometricity by showing that it is decomposable into a sequence of two two-parent geometric crossovers (ii) use twice the simple geometric crossover to produce a symmetric or nearly symmetric three-parent recombination. The second option is indeed very interesting because *it allows us to extended automatically to GPSO all representations we have geometric crossovers for, such as permutations, GP trees, variable-length sequences, to mention few, and virtually any other complex solution representation.*

The resulting generic GPSO algorithm is illustrated in Algorithm 1. This differs from the standard PSO in that: there is no velocity, the equation of position update is the convex combination, there is mutation and the parameters ω, ϕ_1, and ϕ_2 sum up to one. The specific PSO for the Euclidean, Manhattan and Hamming spaces are instances of this general algorithm where the randomized convex combination operators and space-specific mutations are used.

6. EXPERIMENTAL RESULTS

We have run two groups of experiments: one for the continuous version of the GPSO (`EuclideanPSO` or EPSO for short and `ManhattanPSO` or MPSO), and one for the binary version (`HammingPSO`, or HPSO). For the Euclidean and Manhattan versions, we have compared the performances with those of a standard continuous PSO (`BasePSO`, or BPSO) with constriction. We have run the experiments

Algorithm 1 GPSO

for all particle i **do**
  initialise position x_i at random in the search space
end for
while stop criteria not met **do**
  for all particle i **do**
    set personal best x_{p_i} as best position found so far by the particle
    set global best x_g as best position found so far by the whole swarm
  end for
  for all particle i **do**
    update position using a randomized convex combination

$$x_i = CX((x_i, \omega), (x_g, \phi_1), (x_{p_i}, \phi_2)) \quad (2)$$

    mutate x_i
  end for
end while

Table 1: Parameters for continuous versions

κ	0.729
ω	1.0
$\Phi_1 = \Phi_2$	2.05
$V_{max} = X_{max}$	MAX_VALUE - MIN_VALUE
Population Size	20, 50 particles
Stop Condition	200 iterations
Mutation	uniform in [-0.5,0.5]

on the following five benchmark functions: *F1C - Sphere*, *F2C - Rosenbrock*, *F3C - Ackley*, *F4C - Griewangk* and *F5C - Rastrigin* for dimensions 2, 10 and 30. The Hamming version has been tested on the De Jong's test suite: *F1 - Sphere (30)*, *F2 - Rosenbrock (24)*, *F3 - Step (50)*, *F4 - Quartic (240)* and *F5 - Shekel (34)*.

For the continuous versions, we have used the standard PSO parameter set (Table 1), where κ is used only in BPSO and mutation only in EPSO and MPSO. For these two PSOs, ω, Φ_1 and Φ_2 are normalized to sum up to one.

The parameters of population size, number of iterations and ω, Φ_1 and Φ_2 for the binary version have been tuned on the sphere function (see Table 9) and are as in Table 2. From the parameters tuning, it appears that there is a preference for values for ω close to zero. This means that there is a bias towards the swarm and particle bests, and less attraction towards the current particle position.

For each set up we performed 20 independent runs. Table 3 shows the best and the mean fitness value (i.e., the fitness value at the position where the population converges) found by the swarm when exploring continuous spaces. This table summaries the results for the three algorithm presented, over the five test functions, for the two choices of population size, giving an immediate comparison of the performances. Further information is provided in Tables 4–8 where the iteration at which both the best (IBest) and the mean (IMean) values are found are also presented. In all cases the GPSOs, EPSO and MPSO, compare very favourably with BPSO, outperforming it in many cases. This is particularly interesting, since it suggests that the momentum term (not present in GPSO) is not necessary for good performance.

Table 2: Selected parameters for binary version

Population size	100 particles
Iterations	400
Bitwise mutation rate	$1/N$
$\overline{\omega}$	0, 1/12
$\overline{\Phi}_1 = \overline{\Phi}_2$	1/2, 5/12

Table 10 shows the mean of the best fitness value, the iteration when this value is found, the best fitness value over the whole population and the iteration when this value is found for the binary version of the algorithm, HPSO. The algorithm compares well with results reported in the literature, with HPSO obtaining near optimal results on all functions. Interestingly, the algorithm works at its best when ω, the weight for x_i (the particle position), is zero. This corresponds to a degenerated PSO that makes decisions without considering the current position of the particle.

7. CONCLUSIONS AND FUTURE WORK

In [13] we extended the geometric framework by introducing the notion of multi-parent geometric crossover. This is a form of crossover where offspring are in the convex hull of the parents. Then, using the geometric framework, we showed an intimate relation between a simplified form of PSO, without the momentum term, and evolutionary algorithms, which enabled us to generalize in a natural and rigorous way PSO for any type of search space. We specialized the general PSO to Euclidean, Manhattan and Hamming spaces, obtaining three instances of the general PSO for these spaces: EPSO, MPSO and HPSO, respectively. In [13], however, we did not test experimentally the resulting PSOs.

The aim of this paper was to extensively experiment with these new GPSOs. In particular, we applied EPSO, MPSO and HPSO to standard sets of benchmark functions and obtained two surprising results. Firstly, the GPSOs have performed really well, beating the canonical PSO most of the time. Secondly, they have done so right out of the box. That is, unlike the early versions of PSO which required considerable effort before a good general set of parameters could be found, with GPSO we have done very limited preliminary testing and parameter tuning, and yet the new PSOs have worked well. This suggests that they may be quite robust optimisers. This will need to be verified in future research.

An important feature of the GPSO algorithm is that it allows one to automatically define PSOs for all spaces for which a geometric crossover is known [13]. Since geometric crossovers are defined for all of the most frequently used representations and many variations and combinations of those, our geometric framework makes it possible to derive PSOs for all such representations. In future work we will consider other GPSOs, like, for example, for permutation spaces and program spaces.

8. REFERENCES

[1] T. Bäck and D. B. Fogel and T. Michalewicz. Evolutionary Computation 1: Basic Algorithms and Operators. *Institute of Physics Publishing, 2000.*

[2] M. Clerc. Discrete Particle Swarm Optimization, Illustrated by the Traveling Salesman Problem. *New*

Optimization Techniques in Engineering, Springer, 2004.

[3] M. Deza, M. Laurent. Geometry of cuts and metrics. *Springer, 1991.*

[4] J. Kennedy and R.C. Eberhart, *A Discrete Binary Version of the Particle Swarm Algorithm*, IEEE, 1997.

[5] J. Kennedy and R.C. Eberhart, *Swarm Intelligence*, Morgan Kaufmann Publishers, 2001.

[6] S. Luke, L. Spector. A Revised Comparison of Crossover and Mutation in Genetic Programming. *Proceedings of Genetic Programming Conference, 1998.*

[7] A. Moraglio, R. Poli. Topological interpretation of crossover. *Proceedings of the Genetic and Evolutionary Computation Conference, 2004.*

[8] A. Moraglio, R. Poli. Topological crossover for the permutation representation. *GECCO 2005 Workshop on Theory of Representations, 2005.*

[9] A. Moraglio, R. Poli. Geometric Landscape of Homologous Crossover for Syntactic Trees. *Proceedings of the 2005 IEEE Congress on Evolutionary Computation, 2005.*

[10] A. Moraglio, R. Poli, and R. Seehuus. Geometric Crossover for Biological Sequences. *Proceedings of European Conference on Genetic Programming, 2006.*

[11] A. Moraglio, R. Poli. Product Geometric Crossover. *Proceedings of the Parallel Problem Solving from Nature Conference, 2006.*

[12] A. Moraglio, R. Poli. Geometric Crossover for Sets, Multisets and Partitions. *Proceedings of the Parallel Problem Solving from Nature Conference, 2006.*

[13] A. Moraglio, C. Di Chio, and R. Poli. Geometric Particle Swarm Optimisation. *Proceedings of European Conference on Genetic Programming, 2007.*

[14] P. M. Pardalos, M.G.C. Resende. Handbook of Applied Optimization. *Oxford University Press, 2002.*

[15] N. Radcliffe. Equivalence Class Analysis of Genetic Algorithms. *Journal of Complex Systems, 1991*, vol. 5, pages 183–205.

[16] M. L. J. Van de Vel. Theory of Convex Structures. *North-Holland, 1993* .

Table 4: Test results for continuous versions for function F1C - Sphere.

			2	10	30
BPSO	20	Best	-5.35e-14	-1.04	-59.45
		IBest	199	199	199
		Mean	-6.54e-09	-20.75	-168.19
		IMean	199	199	199
	50	Best	-3.67e-13	-0.60	-53.93
		IBest	199	196	190
		Mean	-1.11e-08	-19.09	-176.07
		IMean	199	199	199
EPSO	20	Best	-0.0	-0.0	-0.0
		IBest	145	145	146
		Mean	-0.0	-0.0	-0.0
		IMean	157	158	158
	50	Best	-0.0	-0.0	-0.0
		IBest	141	142	142
		Mean	-0.0	-0.0	-0.0
		IMean	159	160	161
MPSO	20	Best	-0.0	-0.0	-0.0
		IBest	152	168	174
		Mean	-0.0	-0.0	-0.0
		IMean	160	170	175
	50	Best	-0.0	-0.0	-0.0
		IBest	148	165	173
		Mean	-0.0	-0.0	-0.0
		IMean	162	169	174

Table 5: Test results for continuous versions for function F2C - Rosenbrock.

			2	10	30
BPSO	20	Best	-0.00	-36.18	-1912.05
		IBest	199	199	199
		Mean	-97.91	-979.56	-8847.44
		IMean	199	199	199
	50	Best	0.00	-19.46	-1639.46
		IBest	199	199	192
		Mean	-56.04	-791.88	-9425.92
		IMean	199	199	199
EPSO	20	Best	-0.71	-8.98	-28.97
		IBest	3	5	4
		Mean	-1.0	-9.0	-29.0
		IMean	53	44	47
	50	Best	-0.57	-8.96	-28.96
		IBest	1	3	4
		Mean	-1.0	-9.0	-29.0
		IMean	55	52	46
MPSO	20	Best	-0.66	-8.96	-28.97
		IBest	2	3	4
		Mean	-1.0	-9.0	-29.0
		IMean	48	43	45
	50	Best	-0.53	-8.95	-28.95
		IBest	1	3	3
		Mean	-1.0	-9.0	-29.0
		IMean	51	49	46

Table 3: Test results for continuous version: best and mean fitness values found by the swarm over 20 runs at last iteration (iteration 200).

Popsize=20

		BPSO			EPSO			MPSO		
Dim.		2	10	30	2	10	30	2	10	30
F1C	Best	-5.35e-14	-1.04	-59.45	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Mean	-6.54e-09	-20.75	-168.19	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
F2C	Best	-0.00	-36.18	-1912.05	-0.71	-8.98	-28.97	-0.66	-8.96	-28.97
	Mean	-97.91	-979.56	-8847.44	-1.0	-9.0	-29.0	-1.0	-9.0	-29.0
F3C	Best	-3.06e-05	-8.05	-18.09	0.0	0.0	0.0	0.0	0.0	0.0
	Mean	-0.00	-14.86	-20.49	0.0	0.0	0.0	0.0	0.0	0.0
F4C	Best	-0.31	-1.10	-6.67	-0.29	-1.0	-1.0	-0.29	-1.0	-1.0
	Mean	-1.52	-2.98	-17.04	-0.29	-1.0	-1.0	-0.29	-1.0	-1.0
F5C	Best	-0.33	-58.78	-305.11	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Mean	-10.41	-160.98	-504.62	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0

Popsize=50

		BPSO			EPSO			MPSO		
Dim.		2	10	30	2	10	30	2	10	30
F1C	Best	-3.67e-13	-0.60	-53.93	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Mean	-1.11e-08	-19.09	-176.07	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
F2C	Best	0.00	-19.46	-1639.46	-0.57	-8.96	-28.96	-0.53	-8.95	-29.0
	Mean	-56.04	-791.88	-9425.92	-1.0	-9.0	-29.0	-1.0	-9.0	-29.0
F3C	Best	-1.81e-06	-6.78	-17.62	0.0	0.0	0.0	0.0	0.0	0.0
	Mean	-0.00	-15.55	-20.43	0.0	0.0	0.0	0.0	0.0	0.0
F4C	Best	-0.30	-1.05	-6.14	-0.29	-1.0	-1.0	-0.29	-1.0	-1.0
	Mean	-1.63	-2.79	-17.79	-0.29	-1.0	-1.0	-0.29	-1.0	-1.0
F5C	Best	-0.10	-53.67	-302.29	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0
	Mean	-3.56	-159.76	-503.48	-0.0	-0.0	-0.0	-0.0	-0.0	-0.0

Table 10: Test results for HPSO with selected parameters for the De Jong's test suite.

		F1	**F2**	**F3**	**F4**	**F5**
$\overline{\omega} = 0.0$	Best	-0.00015	-0.00034	-0.0	3.45170	-1.13183
	IBest	17	360	49	396	299
	Mean	-5.51540	-54.14453	-2.594	-5.38233	-142.67853
	IMean	399	399	399	399	399
$\overline{\omega} = \frac{1}{6}$	Best	-0.000125	-0.000297	-0.0	3.273980	-1.111220
	IBest	17	360	49	398	179
	Mean	-5.375902	-85.170099	-2.949	-6.919343	-167.283327
	IMean	399	399	399	399	199

Table 6: Test results for continuous versions for function F3C - Ackley.

			2	10	30
BPSO	20	Best	-3.06e-05	-8.05	-18.09
		IBest	199	198	197
		Mean	-0.00	-14.86	-20.49
		IMean	199	199	199
	50	Best	-1.81e-06	-6.78	-17.62
		IBest	199	197	197
		Mean	-0.00	-15.55	-20.43
		IMean	199	199	199
EPSO	20	MeanFit	0.0	0.0	0.0
		IBest	15	15	19
		PopBest	0.0	0.0	0.0
		IMean	20	20	20
	50	MeanFit	0.0	0.0	0.0
		IBest	14	14	14
		PopBest	0.0	0.0	0.0
		IMean	20	21	21
MPSO	20	Best	0.0	0.0	0.0
		IBest	16	18	19
		Mean	0.0	0.0	0.0
		IMean	20	21	20
	50	Best	0.0	0.0	0.0
		IBest	15	18	19
		Mean	0.0	0.0	0.0
		IMean	21	21	21

Table 7: Test results for continuous versions for function F4C - Griewangk.

			2	10	30
BPSO	20	Best	-0.31	-1.10	-6.67
		IBest	187	199	199
		Mean	-1.52	-2.98	-17.04
		IMean	199	199	199
	50	Best	-0.30	-1.05	-6.14
		IBest	162	196	193
		Mean	-1.63	-2.79	-17.79
		IMean	199	199	199
EPSO	20	Best	-0.29	-1.0	-1.0
		IBest	7	6	8
		Mean	-0.29	-1.0	-1.0
		IMean	12	11	11
	50	Best	-0.29	-1.0	-1.0
		IBest	7	5	6
		Mean	-0.29	-1.0	-1.0
		IMean	13	8	12
MPSO	20	Best	-0.29	-1.0	-1.0
		IBest	8	8	11
		Mean	-0.29	-1.0	-1.0
		IMean	12	11	12
	50	Best	-0.29	-1.0	-1.0
		IBest	8	7	11
		Mean	-0.29	-1.0	-1.0
		IMean	13	8	12

Table 8: Test results for continuous versions for function F5C - Rastrigin.

			2	10	30
BPSO	20	Best	-0.33	-58.78	-305.11
		IBest	199	199	182
		Mean	-10.41	-160.98	-504.62
		IMean	199	199	199
	50	Best	-0.10	-53.67	-302.29
		IBest	199	195	189
		Mean	-3.56	-159.76	-503.48
		IMean	199	199	199
EPSO	20	Best	-0.0	-0.0	-0.0
		IBest	8	9	9
		Mean	-0.0	-0.0	-0.0
		IMean	13	12	12
	50	Best	-0.0	-0.0	-0.0
		IBest	8	8	8
		Mean	-0.0	-0.0	-0.0
		IMean	13	12	12
MPSO	20	Best	-0.0	-0.0	-0.0
		IBest	9	11	11
		Mean	-0.0	-0.0	-0.0
		IMean	13	13	13
	50	Best	-0.0	-0.0	-0.0
		IterFit	8	10	11
		Mean	-0.0	-0.0	-0.0
		IMean	13	13	13

Table 9: Test results for HPSO with population size 50 for function F1 - Sphere.

		1/N	2/N	3/N
$\overline{\omega} = 0.0$	Best	-0.000145	-0.000145	-0.00106
	IBest	23	104	193
	Mean	-6.3108868	-8.773445	-14.8852122
	IMean	199	199	199
$\overline{\omega} = \frac{1}{6}$	Best	-0.000145	-0.00015	-0.00179
	IBest	34	129	160
	Mean	-5.9018456	-11.3324751	-14.6928217
	IMean	199	199	199
$\overline{\omega} = \frac{1}{3}$	Best	-0.00011	-0.00023	-0.003535
	IBest	54	190	187
	Mean	-7.553765	-12.3362539	-17.9307488
	IMean	199	199	199
$\overline{\omega} = \frac{2}{3}$	Best	-0.000225	-0.0089	-0.037845
	IBest	181	199	198
	Mean	-12.4479868	-18.4946397	-22.228285
	IMean	199	199	199