Inertial Geometric Particle Swarm Optimization

Alberto Moraglio and Julian Togelius

***Abstract*— Geometric particle swarm optimization (GPSO) is a recently introduced formal generalization of a simplified form of traditional particle swarm optimization (PSO) without the inertia term that applies naturally to both continuous and combinatorial spaces. In this paper, we propose an extension of GPSO, the inertial GPSO (IGPSO), that generalizes the traditional PSO endowed with the full equation of motion of particles to generic search spaces. We then formally derive the specific IGPSO for the Hamming space associated with binary strings and present experimental results for this new algorithm.**

I. Introduction

Particle Swarm Optimization (PSO) is a relatively recently devised popualtion-based stochastic global optimization algorithm [3]. PSO has many similarities with evolutionary algorithms, and has also proven to have robust performance over a variety of difficult optimization problems. However, the original formulation of PSO requires the search space to be continuous and the individuals to be represented as vectors of real numbers.

There are a number of extensions of PSO to combinatorial spaces with various degrees of success [2] [1]. However, every time a new solution representation is considered, the PSO algorithm needs to be rethought and adapted to the new representation.

Geometric Particle Swarm Optimization (GPSO) is a very recently devised generic extension of PSO that can be applied to any search space endowed with a distance and associated with any solution representation to derive formally a specific GPSO for the target space [6]. In previous work, we derived and tested specific GPSOs for different types of continuous spaces and for the Hamming space associated with binary strings [7], for spaces associated with permutations [13] and for spaces associated with Genetic Programming trees [16].

GPSO generalizes a simplified version of PSO in which the inertia coefficient is set to 0, and the sum of the sociality and memory coefficients do not exceed 1. This simplified form of PSO was considered to allow for a simple geometric interpretation of the dynamics of the particles in space which then was used as the basis for the generalization.

When the PSO algorithm was first proposed, it did not include the inertia term. Inertia was introduced in a later development [14]. Inertia turned out to be an essential element to obtain good performances on a variety of problems and became a standard PSO component.

In this paper, we present a formal generalization of the complete PSO equation, including inertia and without restrictions on the coefficients, to general spaces via a geometric interpretation of the dynamic of the particles governed by the full PSO equation. We then formally derive the specific IGPSO for the Hamming space associated with binary strings and present experimental results for this new algorithm.

A. Moraglio is with the Centre for Informatics and Systems of the University of Coimbra, Polo II - University of Coimbra, Coimbra 3030-290, Portugal (email: moraglio@dei.uc.pt).

J. Togelius is with the Dalle Molle Institute for Artificial Intelligence (IDSIA), Galleria 2, Manno-Lugano 6928, Switzerland (email: julian@idsia.ch)

II. The geometry of representations

In this section, we introduce, in an accessible way, the ideas behind a recent formal theory of representations [5] which forms the context for the generalization of the PSO with the full equation presented in the following sections.

Familiar geometric shapes in the Euclidean plane such as for example circles, ellipses, segments, semi-lines, triangles and convex polygons can be defined using distances between points in space. For example, a circle is the locus of points from which the distance to the centre c is a given constant value, the radius r. By replacing in the definition of a shape, say a circle, the Euclidean distance with a different distance, say the Hamming distance, we obtain the definition of a circle in the Hamming space. A circle in the Hamming space looks quite different from a circle in the Euclidean plan, however they both share the same geometric definition. Analogously, if we replace the Euclidean distance with the Manhattan distance, we obtain the definition of a circle in the Manhattan space. A number of simple geometric shapes based on the Manhattan distance in the plane have been derived explicitly (see Taxicab Geometry [4]). We can in fact replace the Euclidean distance in the definition of any geometric shape with any distance meeting a minimum number of requirements (metric), obtaining the corresponding shape in a space with a different geometry. We can also raise the level of abstraction and replace the Euclidean distance with a generic metric, obtaining an abstract shape, such as for example an abstract circle. An abstract circle captures what is common to all circles across all possible geometries. Any property of an abstract circle is also a property of any space-specific circle.

Search algorithms can be viewed from a geometric perspective. The search space is seen as a geometric space with a notion of distance between points, and candidate solutions are points in the space. For example, search spaces associated with combinatorial optimization problems are commonly represented as graphs in which nodes corresponds to candidate solutions and edges between solutions correspond to neighbour candidate solutions. We can endow these spaces with a distance between solutions equal to the length of the shortest path between their corresponding nodes in the graph. Geometric search operators are defined using geometric shapes to delimit the region of search space where to sample offspring solutions relative to the positions of parent solutions. For example, geometric crossover is a search operator

that takes two parent solutions in input corresponding to the end-points of a segment, and returns points sampled at random within the segment as offspring solutions. The specific distance associated with the search space at hand is used in the definition of segment to determine the specific geometric crossover for that space. Therefore, each search space is associated with a different space-specific geometric crossover. However, all geometric crossovers have the same abstract geometric definition.

In analytic geometry, in which points of the Cartesian plane are in one-to-one correspondence with pairs of numbers, their coordinates, the same geometric shape can be equivalently expressed geometrically as a set of points in the plane, or algebraically, by an equation whose solutions are the coordinates of its points. This is an important duality which allows us to treat geometric shapes as equations and vice versa. There is an analogous duality that holds for geometric search operators. Candidate solutions can be thought equivalently as points in space, geometric view, or as a set of syntactic configurations of a certain type, algebraic view. For example, a candidate solution in the Hamming space can be considered as a point in space or as a binary string corresponding to that point. The binary string can then be thought as being the coordinates of the point in the Hamming space. This allows us to think of a search operator equivalently as (i) an algorithmic procedure which manipulates the syntax of the parent solutions to obtain the syntactic configurations of the offspring solutions using well-defined representation-specific operations (algebraic view), or (ii) a geometric description which specifies what points in the space can be returned as offspring for the given parent points and with what probability (geometric view). For example, uniform crossover for binary strings [15] is a recombination operator that produces offspring binary strings by inheriting at each position in the binary string the bit of one parent string or of the other parent string with the same probability. This is an algebraic view of the uniform crossover that tells how to manipulate the parent strings to obtain the offspring string. Equivalently, the same operator can be defined geometrically as the geometric crossover based on the Hamming distance that takes offspring uniformly at random in the segment between parents. There are two important differences between these two definitions of the same operator. The geometric definition is declarative, it defines what offspring the operator returns given their parents without explicitly telling how to actually generate the offspring from the parents. Whereas the algebraic definition is operational, since it defines the search operator by telling for each combination of parents how to build the corresponding offspring. The second important difference is that the geometric description of a search operator is representation-independent and refers only indirectly to the specific solution representation via a distance defined on such representation (i.e. edit distances such as the Hamming distance which can be defined on the binary string representation as the minimum number of bit-flips to obtain one string from the other). In contrast, the algebraic definition of a search operator is representation-dependent and uses operations which are well-defined on the specific solution representation but that may not be well-defined on other representations (e.g. bit-flip on a binary string is not well-defined on a permutation).

The duality of the geometric search operators has surprising and important consequences [5]. The one which is the basis for the present paper is about the possibility of principled generalization of search algorithms for continuous spaces to combinatorial spaces, as sketched in the following. Given a search algorithm defined on continuous spaces, recast the definition of the search operators expressing them explicitly in terms of Euclidean distance between parents and offspring. Substitute the Euclidean distance with a generic metric, obtaining a formal search algorithm generalizing the original algorithm based on the continuous space. Consider a (discrete) representation and a distance associated with it (combinatorial space) and use it in the definition of the formal search algorithm to obtain a specific instance of the algorithm for this space. Use this geometric and declarative description of the search operator to derive its algebraic and operational definition in terms of manipulation of the underlying representation. As mentioned in the introduction, we applied this methodology to generalize a simplified form of PSO to any metric space and derived the specific search operators for a number of representations. In the following sections, we use it to generalize the PSO with the full equation. This methodology can be used to generalize to combinatorial spaces other algorithms naturally based on a notion of distance. This includes search algorithms such as Differential Evolution, Response Surface Methods, Estimation of Distribution Algorithms and Lipschitz Optimization algorithms, but also Supervised Machine Learning algorithms may be generalized to learn functions whose domain and codomain are any types of structured objects. These generalizations are the focus of our current research.

III. Geometric Particle Swarm Optimization

This section prepares the ground for the theory of the generalization of the PSO with inertia, which will be presented in the following section. In this section, we review the theory behind the GPSO algorithm. We first define the concepts of geometric crossover and multi-parent geometric crossover, and use these to define the concept of a convex geometric combination in metric spaces, which is the basis for the generalization of the traditional PSO without inertia. The GPSO algorithm obtained, which can be used independently of its theoretical derivation, is found in section III-D.

A. Geometric crossover

Geometric operators [8] are search operators defined in geometric terms using simple geometric elements such as line segments and balls. These notions and the corresponding genetic operators are well-defined once a notion of distance (metric) in the search space is defined.

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 called the radius of the ball. 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.

Definition 1: 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 really broad. For vectors of reals, various types of blend or line crossovers, box recombinations, and discrete recombinations are geometric crossovers [8]. For binary and multary strings, all homologous crossovers are geometric [8] [10]. For permutations, PMX, Cycle crossover, merge crossover and others are geometric crossovers [11]. For syntactic trees, the family of homologous crossovers are geometric [9]. Recombinations for several more complex representations are also geometric [10] [8] [12].

B. Multi-parent geometric crossover

To extend the geometric crossover to the case of multiple parents we need the notions of metric convex set and metric convex hull.

A set is a metric convex set if for every pair of points within the set, every point in the metric segment that joins them is also within the set.

The metric convex hull of a set of point P is the smallest metric convex set that includes all points in P. For example, in the Euclidean case, when the set P contains two points the convex hull of P is the segment whose extremes are the points in P. When the set P contains three points the convex hull of P is the triangle whose vertices are the points in P.

Definition 2: (Multi-parental geometric crossover) In a multi-parental geometric crossover, given n parents $p_1, p_2, \ldots, p_n$ their offspring are contained in the metric convex hull of the parents $\mathcal{C}(\{p_1, p_2, \ldots, p_n\})$ for some metric d

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

C. Convex combination in metric spaces

In order to define PSO for a generic metric space, we need to extend the notion of convex combination to generic metric spaces.

For the Euclidean space, a *convex combination* is a linear combination of vectors where all coefficients are non-negative and sum up to 1. It is called "convex combination", since, when a vector represents a point in space, all possible convex combinations (given the base vectors) will be within the convex hull of the given points. In fact, the set of all convex combinations constitutes the convex hull.

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. A convex combination for the Euclidean space, not only defines a convex hull, but it tells also how to reach all its points. For convex combinations based on combinatorial spaces, how to achieve this is not obvious. Weighted multi-parental geometric crossovers can be used to pick points specified by convex combinations in combinatorial spaces.

For the Euclidean space, a weighted three-parental geometric crossover can be actually decomposed into two sequential applications of weighted two-geometric crossover: $\Delta GX((a, w_a), (b, w_b), (c, w_c)) = GX((GX((a, \frac{w_a}{w_a+w_b}), (b, \frac{w_b}{w_a+w_b})), w_a + w_b), (c, w_c))$. This formula can be used as a rule of thumb to build weighted three parental geometric crossovers from weighted bi-parental geometric crossovers for any solution representation.

D. GPSO algorithm

Consider the canonical PSO in Algorithm 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.

Theorem 2: In a PSO with no inertia ($\omega = 0$) and where social and memory coefficients 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 $x' = (1 - w_2 - w_3)x + w_2\hat{x} + w_3\hat{g}$.

The generic Geometric PSO algorithm is illustrated in Algorithm 2. This differs from the standard PSO (Algorithm 1) in that: there is no velocity[1], the equation of position

[1]This means that the GPSO algorithm does not keep explicitly track of the velocities of the particles. However, note that every particle that in two successive states is in a different place has a velocity because it has moved of a distance in space in the time unit.

Algorithm 1 Standard PSO algorithm

for all particle i **do**
  initialise position $x_i \in U[\mathbf{a}, \mathbf{b}]$ and velocity $v_i = \mathbf{0}$
end for
while not converged (optimum of current objective function is not found) **do**
  for all particle i **do**
    set personal best $\hat{x}_i$ as best position found so far from the particle (best of current and previous positions)
    set global best $\hat{g}$ as best position found so far from the whole swarm (best of personal bests)
  end for
  for all particle i **do**
    update velocity using equation

$$v_i(t+1) = \omega v_i(t) + \phi_1 R_1(\hat{g}(t) - x_i(t)) + \phi_2 R_2(\hat{x}_i(t) - x_i(t)) \quad (1)$$

    update position using equation

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

  end for
end while

update is the convex combination, there is mutation (to partly compensate for the lack of inertia which would sometimes allow the next position of a particle being generated outside the convex hull) and the parameters ω, ϕ_1, and ϕ_2 are positive and sum up to one. The parameter ω of the GPSO is *not* equivalent to the inertia coefficient in the traditional PSO. This is because the former represents the linear attraction rate toward the previous position of the particle, whereas the latter quantifies the tendency of the new velocity of the particle to be close to its previous velocity in magnitude and direction.

Algorithm 2 Geometric PSO algorithm

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 $\hat{x}_i$ as best position found so far by the particle
    set global best $\hat{g}$ as best position found so far by the whole swarm
  end for
  for all particle i **do**
    update position using a randomized convex combination

$$x_i = CX((x_i, \omega), (\hat{g}, \phi_1), (\hat{x}_i, \phi_2)) \quad (3)$$

    mutate x_i
  end for
end while

IV. Inertial GPSO

A. Geometric Interpretation of the full PSO equation

Let us consider the velocity update equation for the particle i:

$$v_i(t+1) = \omega v_i(t) + \phi_1 R_1(\hat{g}(t) - x_i(t)) + \phi_2 R_2(\hat{x}_i(t) - x_i(t)) \quad (4)$$

where $\omega, \phi_1, \phi_2 \geq 0$ and R_1, R_2 are random variables uniformly distributed in $[0, 1]$.

We can rewrite the equation 4 as follows:

$$v_i(t+1) = k\overline{\omega} v_i(t) + k\overline{\phi_1}(\hat{g}(t) - x_i(t)) + k\overline{\phi_2}(\hat{x}_i(t) - x_i(t)) \quad (5)$$

where $k = \omega + \phi_1 R_1 + \phi_2 R_2, \overline{\omega} = \frac{\omega}{k}, \overline{\phi_1} = \frac{\phi_1 R_1}{k}, \overline{\phi_2} = \frac{\phi_2 R_2}{k}$. We have that $k, \overline{\omega}, \overline{\phi_1}, \overline{\phi_2} \geq 0$ and $\overline{\omega} + \overline{\phi_1} + \overline{\phi_2} = 1$.

From the position update equation (equation 2), we have that the velocities $v_i(t+1), v_i(t)$ in the previous equation can be expressed only in terms of positions as follows:

$$\begin{aligned} v_i(t+1) &= x_i(t+1) - x_i(t) \\ v_i(t) &= x_i(t) - x_i(t-1) \end{aligned} \quad (6)$$

So, replacing them in equation 5 and dividing both sides of the equality by k we obtain:

$$\begin{aligned} \frac{x_i(t+1) - x_i(t)}{k} &= \overline{\omega}(x_i(t) - x_i(t-1)) + \\ &+ \overline{\phi_1}(\hat{g}(t) - x_i(t)) + \overline{\phi_2}(\hat{x}_i(t) - x_i(t)) \end{aligned} \quad (7)$$

We can rewrite equation 7 as follows:

$$\frac{\frac{1}{k} x_i(t+1) + \overline{\omega} x_i(t-1)}{\frac{1}{k} + \overline{\omega}} = \frac{x_i(t)(\overline{\omega} + \frac{1}{k} - \overline{\phi_1} - \overline{\phi_2}) + \overline{\phi_1}\hat{g}(t) + \overline{\phi_2}\hat{x}_i(t)}{\frac{1}{k} + \overline{\omega}} \quad (8)$$

The left-hand side of equation 8 is a convex combination of the vectors $x_i(t+1)$ and $x_i(t-1)$ because their coefficients are larger than 0 ($\overline{\omega}, \frac{1}{k} > 0$) and their sum equals the denominator of the left-hand side of the equation ($\overline{\omega} + \frac{1}{k}$).

The right-hand side of equation 8 is a convex combination of the vectors $x_i(t), \hat{g}(t)$ and $\hat{x}_i(t)$ only when the condition $\overline{\omega} + \frac{1}{k} - \overline{\phi_1} - \overline{\phi_2} > 0$ is met. In this case, since $\overline{\phi_1}, \overline{\phi_2} > 0$, the coefficients of $x_i(t), \hat{g}(t)$ and $\hat{x}_i(t)$ are positive, and their sum equals the denominator of the right-hand side of the equation ($\overline{\omega} + \frac{1}{k}$).

If the condition that makes the right-hand side of equation 8 a convex combination is not satisfied, we can rewrite equation 8 as follows:

$$\frac{\frac{1}{k} x_i(t+1) + \overline{\omega} x_i(t-1) + x_i(t)(\overline{\phi_1} + \overline{\phi_2})}{\frac{1}{k} + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2}} = \frac{x_i(t)(\overline{\omega} + \frac{1}{k}) + \overline{\phi_1}\hat{g}(t) + \overline{\phi_2}\hat{x}_i(t)}{\frac{1}{k} + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2}} \quad (9)$$

Both sides of the equation 9 are convex combinations, without requiring any extra condition.

B. Geometric generalization of the PSO equations

Equations 8 and 9 describe exactly the same dynamics of the complete PSO equation. Furthermore, they have a direct geometric interpretation that allows us to generalize them to any metric space.

For each particle i of the swarm, at a given time t, we know the positions $x_i(t-1), x_i(t), \hat{x}_i(t)$ and $\hat{g}(t)$. Also, we know all coefficients of the convex combinations involved in the equations 8 and 9 which are (constant) parameters which can be derived from the parameters ω (inertia), ϕ_1 (sociality) and ϕ_2 (memory) of the PSO. Therefore, at each time, we need to compute the new position $x_i(t+1)$ of the particle i. In sections IV-B.1, IV-B.2 and IV-B.3, we present the geometric constructions to determine $x_i(t+1)$ for the GPSO (figure 1) and for the IGPSO based on equations 8 and 9 (figures 2 and 3).

In order to obtain the generalization to metric spaces, beside a well-defined notion of convex combination, we need a well-defined notion of extension ray. The extension ray $ER(A, B)$ in the Euclidean plane is a semi-line originating in A and passing through B (note that $ER(A, B) \neq ER(B, A)$). The extension ray can be defined using only

distances indirectly using metric segments: $C \in ER(A, B)$ iff $C \in [A, B]$ or $B \in [A, C]$. So, it is well-defined for any metric space and any representation. In our case, only the part of the extension ray beyond B will be of interest because the point C that we want to determine is never between A and B by construction.

We can define a notion of weighted extension ray recombination $C = ER((A, w_{ab}), (B, w_{bc}))$ as the inverse operation of weighted convex combination CX of two points A and C, $B = CX((A, w_{ab}), (C, w_{bc}))$, as follows. The distances $d(A, B)$ and $d(B, C)$ are proportional to the weights w_{ab} and w_{bc}, which are positive real number between 0 and 1 and sum up to 1. In a weighted extension ray recombination, we are given A, B, w_{ab} and w_{bc}, and we want to determine C. We can compute the distance $d(B, C) = d(A, B) * w_{bc}/w_{ab}$. Then return those points C which are at a distance $d(A, B) + d(B, C)$ from A *and* at a distance $d(B, C)$ from B.

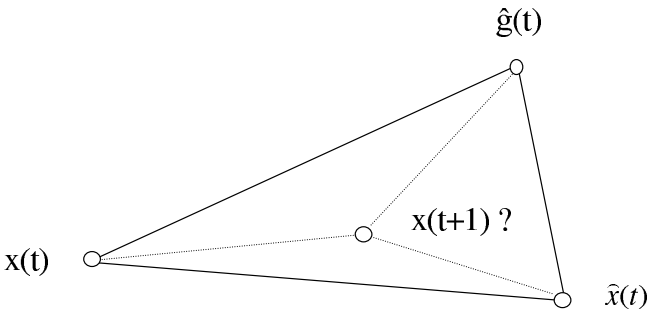

Fig. 1. Geometric PSO

1) Geometric construction for GPSO: In GPSO, we can determine $x(t + 1)$ (see figure 1) by the weighted convex combination $CX(x(t), \hat{g}(t), \hat{x}(t))$ with weights $w_1 = 1 - \phi_1 + \phi_2$, $w_2 = \phi_1$ and $w_3 = \phi_2$. Note that $\phi_1 + \phi_2 \leq 1$ and after the convex combination, $x(t + 1)$ undergoes mutation.

2) Geometric construction for IGPSO - simple case: In IGPSO as in traditional PSO, we have 3 parameters: ω, ϕ_1 and ϕ_2, and no mutation. These parameters must be non-negative and can be larger than 1 (as in the tarditional PSO).

Figure 2 illustrates the geometric relation between points described by equation 8. Geometrically, the equation we need to solve to determine $x(t + 1)$ is $CX(x(t - 1), x(t + 1)) = CX(x(t), \hat{g}(t), \hat{x}(t))$. Operationally, this can be done in two steps, one convex combination of three points, followed by one extension ray with two points:

1) $e(t) = CX(x(t), \hat{g}(t), \hat{x}(t))$
2) $x(t + 1) = ER(x(t - 1), e(t))$

The weights of CX and ER are functions of the parameters ω, ϕ_1 and ϕ_2, and correspond to the coefficients of equation 8. For CX we have:

$w_1 = (\overline{\omega} + 1/k - \overline{\phi_1} - \overline{\phi_2})/(1/k + \overline{\omega})$
$w_2 = \overline{\phi_1}/(1/k + \overline{\omega})$
$w_3 = \overline{\phi_2}/(1/k + \overline{\omega})$

And for ER we have:

$w_1 = \overline{\omega}/(1/k + \overline{\omega})$
$w_2 = 1/k/(1/k + \overline{\omega})$

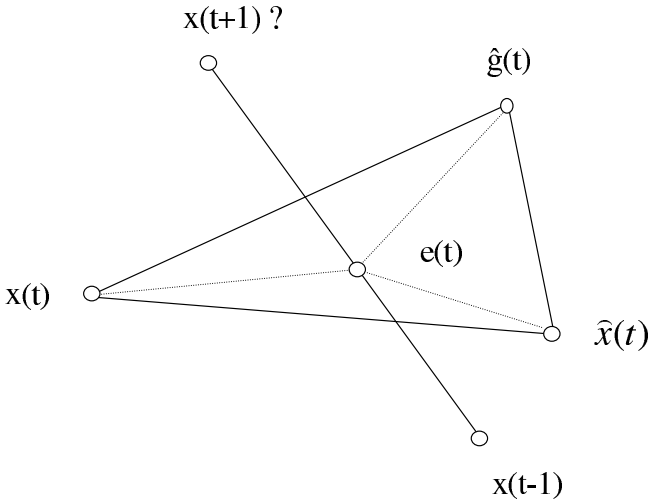

Fig. 2. Inertial Geometric PSO - simple case

where $k = \omega + \phi_1 \cdot R_1 + \phi_2 \cdot R_2$, $\overline{\omega} = \omega/k$, $\overline{\phi_1} = \phi_1 \cdot R_1/k$ and $\overline{\phi_2} = \phi_2 \cdot R_2/k$.

In the traditional PSO, R_1 and R_2 are random numbers uniformly distributed in $[0, 1]$. However, in IGPSO the same effect of randomizing the coefficients using random numbers R_1 and R_2 is achieved by the randomization of the convex hull and extension ray operators. To preserve as much as possible the same interpretation for both traditional PSO and for the IGPSO of the coefficients ω, ϕ_1 and ϕ_2, we set $R_1 = 0.5$ and $R_2 = 0.5$ to the average values of their distribution.

The simple case of geometric construction presented in this section holds provided the condition $\omega + 1 > \phi_1 + \phi_2$ is met. This condition derives from the condition $\overline{\omega} + 1/k - \overline{\phi_1} - \overline{\phi_2} > 0$ associated with equation 8. If this condition is not met, we need to use the recombination which applies to the general case with no restrictions on the coefficients of the PSO. The general recombination degenerates to the simple case when the above condition is met.

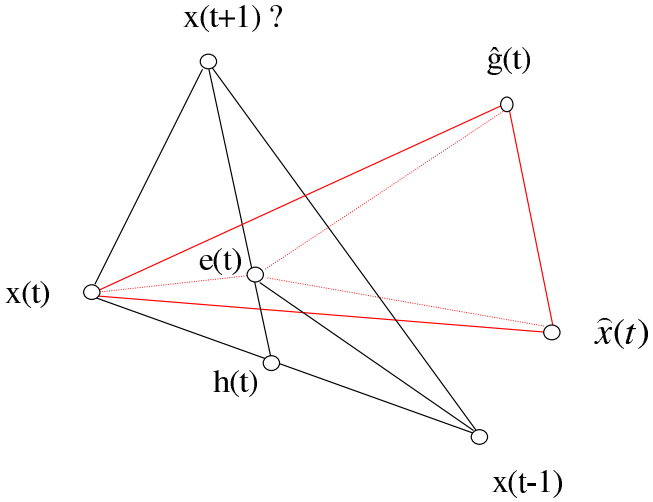

Fig. 3. Inertial Geometric PSO - general case

3) Geometric construction for IGPSO - general case: Figure 3 illustrates the geometric relation between points described by equation 9. Geometrically, the equation we need to solve to determine $x(t+1)$ is $CX(x(t), x(t-1), x(t+1)) = CX(x(t), \hat{g}(t), \hat{x}(t))$. Operationally, this can be done in three steps, two convex combinations and one extension ray recombination:

1) $e(t) = CX1(x(t), \hat{g}(t), \hat{x}(t))$
2) $h(t) = CX2(x(t), x(t-1))$
3) $x(t+1) = ER(h(t), e(t))$

From the right-hand side of equation 9, the weights for CX1 are:

$w_{11} = (\overline{\omega} + 1/k)/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$
$w_{12} = \overline{\phi_1}/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$
$w_{13} = \overline{\phi_2}/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$

where k, $\overline{\omega}$, $\overline{\phi_1}$ and $\overline{\phi_2}$ are defined as in the simple case.

To determine the weights of CX2 and ER, we can reason as follows. If we knew $x(t+1)$, from the left-hand side of equation 9 the weights to determine $e(t)$ by a convex combination $CX3(x(t), x(t-1), x(t+1))$ would be:

$w_{31} = (\overline{\phi_1} + \overline{\phi_2})/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$
$w_{32} = \overline{\omega}/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$
$w_{33} = (1/k)/(1/k + \overline{\omega} + \overline{\phi_1} + \overline{\phi_2})$

Let us now decompose the convex combination CX3 in two successive applications of a 2-parental recombination RX, using the decomposition formula:

$CX((a, w_a), (b, w_b), (c, w_c)) = RX((d, w_d), (c, w_c))$

where $d = RX((a, w_a/(w_a + w_b)), (b, w_b/(w_a + w_b)))$ and $w_d = w_a + w_b$.

Applying it to CX3, we have:

$CX((x(t), w_{31}), (x(t-1), w_{32}), (x(t+1), w_{33})) = RX1((h, w_h), (x(t+1), w_{33}))$

where $h = RX2((x(t), w_{31}/(w_{31} + w_{32})), (x(t-1), w_{32}/(w_{31} + w_{32})))$ and $w_h = w_{31} + w_{32}$

From this decomposition, the weights for $h(t) = CX2(x(t), x(t-1))$ are the same as for $RX2$:

$w_{21} = w_{31}/(w_{31} + w_{32})$
$w_{22} = w_{32}/(w_{31} + w_{32})$

and the weights for $x(t+1) = ER(h(t), e(t))$ are the same as for $RX1$:

$w_1 = w_h = w_{31} + w_{32}$
$w_2 = w_{33}$

The pseudocode for the IGPSO is given in algorithm 3.

Algorithm 3 Inertial Geometric PSO algorithm

```
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 x̂_i to best position found so far by the particle
        set global best ĝ to best position found so far by the whole swarm
    end for
    for all particle i do
        update position using a randomized convex combination
        update position using weighted extension ray
    end for
end while
```

V. IGPSO for the Hamming space

In previous work, we have already presented a weighted convex combination for the Hamming space [6]. This turned out to be equivalent to a 3-parental uniform crossover for binary strings. In the following, we derive the weighted extension ray recombination for the Hamming space.

A. Extension ray in the Hamming space

The extension ray is defined as: $C \in ER(A, B)$ iff $C \in [A, B]$ or $B \in [A, C]$. Let us consider an example of extension ray in the Hamming space. Let $A = 110011$ and $B = 111001$.

The relation $C \in [A, B]$ is satisfied by those C that match the schema $S1 = 11 * 0 * 1$. This is because: (i) this is the set of the possible offspring of A and B that can be obtained by recombining them using the uniform crossover, and (ii) this operator corresponds to the uniform geometric crossover under Hamming distance which returns offspring in the segment between parents.

The relation $B \in [A, C]$ is satisfied by all those C that when recombined with A using the uniform crossover can produce B as offspring. This set, in turn, corresponds to those C that match $S2 = **1*0*$, as we explain in the following.

The schema $S1$ is obtained by keeping the common bits in A and B and inserting $*$ where the bits of A and B do not match. The schema $S2$ is obtained by inserting $*$ where the bits are common in A and B and inserting the bits coming from B where the bits of A and B do not match. All C matching $S2$ recombined with A can produce B as offspring. This is because at each position (in A, B and C) when in the schema $S2$ there is a star the bit in B at that position can be inherited from A. When in the schema there is a bit (0 or 1) the bit in B at that position can be inherited from C. Notice also that only the strings C matching $S2$ can produce B when C is recombined with A.

B. Weighted extension ray recombination in the Hamming space

In a weighted extension ray recombination, we are given A, B, w_{ab} and w_{bc}, and we want to determine C. As mentioned earlier, we can compute the distance $d(B, C) = d(A, B) \cdot w_{bc}/w_{ab}$. Then return those points C which are at a distance $d(A, B) + d(B, C)$ from A *and* at a distance $d(B, C)$ from B. Notice that, since in the Hamming space the maximum distance between two strings in the space is bounded by n (number of bits), there is an upper-bound to $d(B, C)$ which has to be (forced to be) $\leq n - d(A, B)$.

Definition 3: The extension ray recombination of A and B is as follows. Let us define the probability $P = d(B, C)/(n - d(A, B))$. For each position, if A and B have the same bit, with probability P put the complementary bit in C at that position. If A and B have different bits, put the bit of B in C at that position.

Theorem 3: Using this recombination the expected distance between B and the generated C, $\mathbf{E}[d(B, C)]$, over the distance $d(A, B)$ equals the ratio w_{bc}/w_{ab}. Therefore,

this recombination operator fits the geometric definition of weighted extension ray under Hamming distance.

Proof: This can be shown as follows. The number of bits in which A and B differ are $d(A, B)$. The number of bits in which A and B do not differ is $n - d(A, B)$. For the bits in which A and B differ, the string C equals B. For each bit in which A and B do not differ, C does not equal B with probability P. So, the expected distance between B and C is $\mathbf{E}[d(B, C)] = (n - d(A, B)) \cdot P$. By substituting $P = d(B, C)/(n - d(A, B))$, we have $\mathbf{E}[d(B, C)] = d(B, C) = d(A, B) \cdot w_{bc}/w_{ab}$. So, $\mathbf{E}[d(B, C)]/d(A, B) = w_{bc}/w_{ab}$. ■

Notice that the extension ray recombination operator differs from bitwise mutation with probability P applied to B in that in the former operator not every bit has probability P of undergoing mutation (only those bits in which B equals A). Also, the extension ray recombination differs from a combined operator of crossover of A and B followed by a bitwise mutation of the offspring. In particular, the latter operator may generate offspring beyond A with respect to B (in the extension ray with origin B and passing through A). This cannot happen with the extension ray recombination with origin A and passing through B.

VI. Experiments

We implemented the IGPSO algorithm for binary spaces within a Java framework,[2] and investigated its performance on some benchmark problems. The proposed algorithm was compared with the GPSO algorithm and with three other algorithms:

- BPSO: Discrete Binary PSO of Kennedy and Eberhart, using the results presented in [2].
- GA: A canonical Genetic Algorithm, with roulette wheel fitness-proportionate selection, uniform crossover and bitflip mutation.
- ES: A $\mu + \lambda$ Evolution Strategy, with $\mu = \lambda = popsize/2$ and bitflip mutation.

For the two latter algorithms, the bitflip mutation works as follows: each bit in the chromosome is considered, and with probability p this bit is flipped. In the experiments involving these algorithms, this parameter was systematically varied between 0.0 and 0.5 in increments of 0.01. For the IGPSO experiments, the key parameters ω, ϕ_1 and ϕ_2 were systematically varied between 0.0 and 2.0 in increments of 0.25. For GPSO, ω and ϕ_1 were varied between 0.0 and 1.0 in increments of 0.1.

In all experiments, the length of any single run was set to 4000 function evaluations, in order to be directly comparable with the results of Kennedy and Eberhart. For IGPSO, GPSO, GA and ES the population size was varied systematically: sizes of 10, 20, 40, 80 and 160 were tried, with the numbers of generations limited appropriately: 400, 200, 100, 50 and 25.

We used three of the same benchmark problems that Kennedy and Eberhart tested their algorithm on. These are William Spears' binary versions of DeJong's functions $f1$, $f2$ and $f3$[3]. (We did not use $f4$ and $f5$ due to unresolved differences between different versions of the code, which might be due to differing numerical precision in different systems; further, Kennedy and Eberhart do not report precise results for $f4$.)

[2]Source code is available upon request from the second author.

Algorithm	$f1$ (78.6)		$f2$ (3905.93)		$f3$ (55.0)	
GPSO	78.5996	5	3905.9284	2	55.0	20
IGPSO	78.5998	6	3905.9289	2	55.0	20
BPSO	-	10	-	4	-	20
GA	78.2152	0	3905.8052	0	52.1	1
ES	78.5998	7	3905.9291	2	55.0	20

TABLE I

Results of the empirical experiments. The maximum of the functions are reported next to their names. For each combination of algorithm and problem, the results of the best parameterization of that combination are reported. The first number is the best fitness of the last generation, averaged over 20 runs. The second number is the number of those runs that reached the global optimum.

GPSO	pop/gen	mut	ω	ϕ_1	ϕ_2
$f1$	80/50	0.2	0.0	0.5	0.5
$f2$	80/50	0.1	0.5	0.0	0.5
$f3$	*	*	*	*	*
IGPSO	pop/gen	mut	ω	ϕ_1	ϕ_2
$f1$	20/200	N/A	1.75	1.25	0.5
$f2$	40/100	N/A	1.75	1.25	1.25
$f3$	*	N/A	*	*	*

TABLE II

Best parameter settings found for GPSO and IGPSO. The asterisks denote that many combinations are optimal.

Each configuration (parameters and population size) was tested twenty times, and the average best score of each run was recorded, as well as how many of the runs that reached the global optimum. The results are summarized in table I. The parameters were optimized separately for each combination of benchmark function and algorithm, and only the results of the best configuration are reported here. The best parameter settings found are reported in tables II and III.

The very compressed fitness structure of $f1$ and $f2$, with many local optima with values differing from the global optimum only in the third decimal, is a very bad match with fitness-proportional selection; the landscape of $f3$ has similar characteristics but to a lesser degree. Therefore, the results of the GA are by far the worst on all problems.

In comparison, the GA always works best with large populations and relatively high mutation rates (> 0.1). The ES seems to be relatively insensitive to population size, as long as the mutation rate is in the region 0.05–0.1.

The results indicate that GPSO and IGPSO perform very similarly on the tested problems. From table I we can see that both algorithms perform similarly to a standard ES, much better than the GA, and are competitive with BPSO on

[3]The original c source code of these functions can be found at http://www.cs.uwyo.edu/ wspears/functs/dejong.c

GA	pop/gen	mutation
$f1$	160/25	0.12
$f2$	160/25	0.16
$f3$	80/50	0.39
ES	pop/gen	mutation
$f1$	10/400	0.13
$f2$	160/25	0.1
$f3$	*	*

TABLE III

BEST PARAMETER SETTINGS FOUND FOR GA AND ES. THE ASTERISKS DENOTE THAT MANY COMBINATIONS ARE OPTIMAL.

these particular problems. It is a bit surprising that IGPSO does not perform better than GPSO, which it is an extension of. A likely explanation is that GPSO includes a mutation operator, which IGPSO does not. Mutation might provide the same exploration benefit as inertia implemented through movement along the extension ray does (for IGPSO) on these particular problems; indeed, the most successful parameter settings found for GPSO includes significant mutation. Another part of the explanation might be that the particular binary encoding that is used in these problems induces a search space where inertia is of little use. It remains to be seen whether the benefits of inertia will become clearer when testing the algorithm on other problems.

VII. CONCLUSION

We have introduced a principled generalization of the complete standard particle swarm optimization algorithm to arbitrary spaces: Inertial Geometric Particle Swarm Optimization, or IGPSO. This algorithm extends the previously proposed GPSO algorithm by also taking inertia into account, by means of a weighted extension ray. Further, we have described the construction of such an extension ray in Hamming space. Initial results of applying IGPSO to a classic benchmark are promising. In the future, further studies should be carried out on other benchmark functions, and in other combinatorial spaces.

REFERENCES

[1] M. Clerc, *Discrete particle swarm optimization, illustrated by the traveling salesman problem*, New Optimization Techniques in Engineering, Springer, 2004, pp. 219–239.

[2] J. Kennedy and R. C. Eberhart, *A discrete binary version of the particle swarm algorithm*, IEEE Transactions on Systems, Man, and Cybernetics **5** (1997), 4104–4108.

[3] ———, *Swarm intelligence*, Morgan Kaufmann, 2001.

[4] Eugene F. Krause, *Taxicab geometry: An adventure in non-euclidean geometry*, Courier Dover Publications, 1986.

[5] A. Moraglio, *Towards a geometric unification of evolutionary algorithms*, Ph.D. thesis, University of Essex, 2007.

[6] A. Moraglio, C. Di Chio, and R. Poli, *Geometric particle swarm optimization*, European Conference on Genetic Programming, 2007, pp. 125–136.

[7] A. Moraglio, C. Di Chio, J. Togelius, and R. Poli, *Geometric particle swarm optimization*, Journal of Artificial Evolution and Applications **2008** (2008), Article ID 143624.

[8] A. Moraglio and R. Poli, *Topological interpretation of crossover*, Proceedings of the Genetic and Evolutionary Computation Conference, 2004, pp. 1377–1388.

[9] ———, *Geometric landscape of homologous crossover for syntactic trees*, Proceedings of IEEE congress on evolutionary computation, 2005, pp. 427–434.

[10] ———, *Product geometric crossover*, Proceedings of Parallel Problem Solving from Nature conference, 2006, pp. 1018–1027.

[11] ———, *Topological crossover for the permutation representation*, Journal of the Italian Association for Artificial Intelligence (2007), (to appear).

[12] A. Moraglio, R. Poli, and R. Seehuus, *Geometric crossover for biological sequences*, Proceedings of the European Conference on Genetic Programming, 2006, pp. 121–132.

[13] A. Moraglio and J. Togelius, *Geometric pso for the sudoku puzzle*, Proceedings of the Genetic and Evolutionary Computation Conference, 2007, pp. 118–125.

[14] Y.H. Shi and R.C. Eberhart, *A modified particle swarm optimizer*, IEEE International Conference on Evolutionary Computation, 1998.

[15] G. Sywerda, *Uniform crossover in genetic algorithms*, Proceedings of the third international conference on Genetic algorithms, 1989.

[16] Julian Togelius, Renzo De Nardi, and Alberto Moraglio, *Geometric pso + gp = particle swarm programming*, Proceedings of the Congress on Evolutionary Comptutation (CEC), 2008.