mathematics

MDPI

Article

Memes Evolution in a Memetic Variant of Particle Swarm Optimization

Umberto Bartoccini [1], Arturo Carpi [2], Valentina Poggioni [2] and Valentino Santucci [1,*]

[1] Department of Humanities and Social Sciences, University for Foreigners of Perugia, 06123 Perugia, Italy; umberto.bartoccini@unistrapg.it

[2] Department of Mathematics and Computer Science, University of Perugia, 1-06121 Perugia, Italy; arturo.carpi@unipg.it (A.C.); valentina.poggioni@unipg.it (V.P.)

* Correspondence: valentino.santucci@unistrapg.it

Received: 1 April 2019; Accepted: 7 May 2019; Published: 11 May 2019

Abstract: In this work, a coevolving memetic particle swarm optimization (CoMPSO) algorithm is presented. CoMPSO introduces the memetic evolution of local search operators in particle swarm optimization (PSO) continuous/discrete hybrid search spaces. The proposed solution allows one to overcome the rigidity of uniform local search strategies when applied to PSO. The key contribution is that memes provides each particle of a PSO scheme with the ability to adapt its exploration dynamics to the local characteristics of the search space landscape. The objective is obtained by an original hybrid continuous/discrete meme representation and a probabilistic co-evolving PSO scheme for discrete, continuous, or hybrid spaces. The coevolving memetic PSO evolves both the solutions and their associated memes, i.e. the local search operators. The proposed CoMPSO approach has been experimented on a standard suite of numerical optimization benchmark problems. Preliminary experimental results show that CoMPSO is competitive with respect to standard PSO and other memetic PSO schemes in literature, and its a promising starting point for further research in adaptive PSO local search operators.

Keywords: memetic particle swarm optimization; adaptive local search operator; co-evolution; particle swarm optimization; PSO

1. Introduction

Memetic algorithms (MAs) are a class of hybrid evolutionary algorithms (EAs) and have been widely applied to many complex optimization problems [1], such as scheduling problems [2–4], combinatorial optimization problems [5–7], multi-objective optimization problems [8,9], and multimodal optimization problems [10,11]. Population-based evolutionary approaches, such as genetic algorithms (GAs) or particle swarm optimization (PSO) [12], are able to detect in a fast way the main regions of attraction, while their local search abilities represent a major drawback [13] from the point of view of solution accuracy and of the convergence behaviour, especially when applied to multimodal problems. MAs [1,14] have recently received attention as effective meta-heuristics to improve generic EA schemes by combining these latter with local search (LS) procedures [13,15]. In MAs, EA operations are employed for global rough exploration, and LS operators are used to execute further exploitation of single EA individuals. This allows one to enhance the EA's capability to solve complicated problems. In the context of PSO [12], the LS procedure can be regarded as an iterative local move of the particle. The scheme has been successfully applied to improve different meta-heuristics such as simulated annealing [16], GAs [17], and PSO [18]. A memetic algorithm that hybridizes PSO with LS, called MPSO, is proposed in [11] for locating multiple global and local optima in the fitness landscape of multimodal optimization problems (MMOPs). In this work, the local search is designed by means of two different operators,

cognition-based local search (CBLS) and random walk with direction exploitation (RWDE), but no adaptation technique of the local search operators is employed.

Although many GA-based MAs and other EA approaches employing meme evolution have been proposed in the literature [13,19], the PSO memetic algorithms proposed so far fail to employ meme evolution, probably because of the difficulty of designing the descriptors of the local search operators, which are usually statically tailored to the problem, or the class of problems, at hand [7,8,10,11,20]. Indeed, they exhibit, at most, only some form of temporary adaptation that is not preserved or transmitted through the generations as meta-Lamarkian memes [21].

Since meme evolution is able to provide adaptivity of the local search operators, and many research results have shown that the advantages of the introduction of a local search in EAs, namely in PSO, are generalized, we have set as a research objective the investigation of the possibility of introducing memetic evolution of LS operators in a PSO context. The idea was to design a framework where memes are not selected in a preliminary evolutionary phase, but where they evolve and adapt during the search.

The research resulted in the coevolving memetic particle swarm optimization (CoMPSO) algorithm presented in this work. CoMPSO is an algorithm for numerical optimization in a hybrid sea and allows one to evolve local search operators in the form of memes associated with the PSO particles exploring a continuous and/or discrete hybrid search space. A special feature of local search operators, defined by the memes, is that they are applied in a probabilistic way. A meme is applied to the personal best position of the associated particle with a given probability and every given number of iterations. The basic idea is that memes are evolved themselves by means of the PSO-MEME evolution scheme, a classical PSO scheme modified to manage meme vectors composed of hybrid continuous/discrete components. Indeed, the PSO-MEME scheme distinguishes the evolution of continuous and discrete components where a probabilistic technique is applied.

The rest of the paper is structured as follows. Some general issues concerning MAs and PSO are recalled in Section 2.1, while the coevolving memetic PSO scheme is introduced in Section 2.2 by describing the proposed co-evolving particle-meme scheme and a probabilistic technique for PSO evolution of discrete integer components. Experimental results and comparisons with classical PSO and state-of-the-art static memetic PSO are presented and discussed in Section 3. Finally, in Section 4, a discussion of the advantages of the proposed approach and a description of future lines of research conclude the paper.

2. Co-Evolution of Memes in PSO: Models and Methods

In this section, we introduce the architecture of CoMPSO by first recalling some concepts of memetic algorithms and PSO, respectively in Sections 2.1.1 and 2.1.2. A detailed description of the CoMPSO scheme is then provided in Section 2.2, focusing on the algorithmic aspects introduced to guarantee diversity and on the representation for memes in the discrete, continuous, and hybrid space search, and finally describing and explaining the motivations for meme co-evolution strategies.

2.1. Memetic Algorithms and Particle Swarm Optimization

2.1.1. Memetic Algorithms

MAs, first proposed by Moscato [14,22], represent a recent growing area of research in evolutionary computation. The MA meta-heuristics essentially combine an EA with local search techniques that can be seen as a learning procedure that makes individuals capable of performing local refinements. MAs take inspiration from the Dawkins notion of "meme" [23] representing a unit of cultural evolution that can exhibit refinement.

MAs are usually distinguished into three main classes. First generation MAs [22] are characterized by a single fixed meme; i.e., a given local search operator is applied to candidate solutions of the main EA according to different selection and application criteria (see, for example, [20]). The criteria

used in the selection of candidates and the frequency of the local search application represent the parameters of MAs. Second generation MAs, also called "meta-Lamarckian MAs" [21,24], reflect the principle of memetic transmission and selection; i.e., they are characterized by a pool of given local search operators (memes) that compete, based on their past merits, to be selected for future local refinements. The choice of the best memes can be based on different criteria, such as the absolute fitness of the candidate solutions associated with the meme or the fitness improvement due to past applications. Finally, the most recent generation of MA schemes are the co-evolving MAs [10,15,25,26], which introduce the idea of meme evolution. In this case, the pool of memes is not known a priori, but the memes are represented and evolved by means of EA methods similar to those applied for candidate solutions. The CoMPSO scheme proposed in this paper falls in this latter class.

2.1.2. Particle Swarm Optimization

Particle swarm optimization (PSO), originally introduced in [18], is a meta-heuristic approach to continuous optimization problems that is inspired by the collective behavior observed in flocks of birds. Further variants of PSO have been proposed in [12,27–33].

In PSO, a swarm of artificial entities—the so called *particles*—navigates the search space, aiming at optimizing a given objective/fitness function $f : \Theta \to \mathbb{R}$, where $\Theta \subseteq \mathbb{R}^d$ is the feasible region of the space. The set $P = \{p_1, \ldots, p_n\}$, composed of n particles, is endowed with a neighborhood structure; i.e., a set $N_i \subseteq P$ of neighbors is defined for each particle p_i. Every particle has a position in the search space, which is iteratively evolved using its own search experience and the experience of its neighbors. Therefore, PSO adopts both cognitive and social strategies [34] in order to focus the search on the most promising areas.

At every iteration $t \in \mathbb{N}$, any particle p_i is composed of the following d-dimensional vectors: the *position* in the search space $x_{i,t}$, the *velocity* $v_{i,t}$, the *personal best* position $b_{i,t}$ visited so far by the particle, and the *neighborhood best position* $g_{i,t}$ visited so far among the particles in N_i.

Both positions and velocities of the PSO particles are randomly initialized. Then they are iteratively updated until a stop criterion is met (e.g., the allowed budget of fitness evaluations has been consumed), according to the following *move equations* (first studied in [18]):

$$v_{i,t+1} = \omega v_{i,t} + \varphi_1 r_{1,t}(b_{i,t} - x_{i,t}) + \varphi_2 r_{2,t}(g_{i,t} - x_{i,t}), \tag{1}$$

$$x_{i,t+1} = x_{i,t} + v_{i,t+1}. \tag{2}$$

In Equation (1), the weight ω is the inertial coefficient, φ_1 and φ_2 are the acceleration factors, while the two random vectors $r_{1,t}, r_{2,t} \in \mathbb{R}^d$ are uniformly distributed in $[0,1]^d$. The first term of Equation (1)—the *inertia* or momentum—is a memory of the trajectory followed by the particle so far. Its aim is to prevent drastic changes in the search performed by the particle. The second term is the *cognitive component* and represents the tendency of the particle to return to the best position ever visited by itself. Finally, the third term—the *social component*—models the contribution of the neighbors in the particle's trajectory.

After every position update, the fitness of $x_{i,t}$ is evaluated. If it is better than those of the previous personal and social best positions, these are updated accordingly.

Usually a fully connected topology is adopted as a neighborhood graph among the particles, so only one global social best g_t is needed instead of n local copies of it. It should be noted that sometimes the particles can exceed the bounds of the feasible search space. A common practice to address this issue, adopted also in our work, is to restart the out-of-bounds components of a particle in a position randomly chosen between the old position and the exceeded bound [35,36].

2.2. Coevolving Memetic PSO

Coevolving memetic PSO (CoMPSO) is a hybrid algorithm that combines PSO with the evolution of local search memetic operators that automatically adapt to the objective function landscape.

CoMPSO evolution can be seen as the combination of two co-evolving populations: the particles and the memes. The PSO particles represent the candidate solutions and evolve by means of the usual PSO laws (see Section 2.1.2). The memes represent the local search operators applicable to particles and are evolved by a modified PSO scheme that employs an innovative technique designed to deal with the discrete domains present in the meme representation. Furthermore, another difference with respect to the standard PSO scheme is the introduction of a diversity control mechanism that prevents the premature convergence of the particle population to a local optimum. Other example of the hybridization of PSO can be found in the area of pattern recognition of strings [37–40].

In the following, the general CoMPSO scheme (Section 2.2.1), the diversity control mechanism of particles evolution (Section 2.2.2), the meme representation (Section 2.2.3), meme evolution (Section 2.2.4), and the novel technique for managing discrete components in meme PSO (Section 2.2.5) are introduced.

2.2.1. The CoMPSO Scheme

In the CoMPSO general scheme, a population of n particles, i.e., $P = \{p_1, \ldots, p_n\}$, navigates the search space following PSO dynamics while trying to find the optimum of the given fitness function f. At each iteration, a local search is possibly applied to a subset of particles in order to improve their fitness. This local search phase is realized by a meme population, i.e., $M = \{m_1, \ldots, m_n\}$, of local search operators. Each m_i is associated with particle p_i and is evolved using a PSO-like approach on the meme space.

Particles in P are encoded using the usual d-dimensional vectors of the PSO scheme (see Section 2.1.2), i.e., the position vector $x_{i,t}^{(p)}$, the velocity vector $v_{i,t}^{(p)}$, and the personal best vector $b_{i,t}^{(p)}$, other than the common global best $g_t^{(p)}$. In this work, a complete neighborhood graph has been adopted. Similarly, a generic meme $m_j \in M$ is encoded by the hybrid vectors $x_{j,t}^{(m)}$, $v_{j,t}^{(m)}$, $b_{j,t}^{(m)}$, and $g_t^{(m)}$, i.e., vectors that combine both discrete and continuous components as described in Section 2.2.5.

The general CoMPSO scheme is described by the pseudo-code reported in Algorithm 1.

The populations of particles P and memes M are randomly initialized in their respective feasible regions. During main-loop iterations, each particle p_i is evolved following the scheme described in Section 2.1.2. *LOCAL_SEARCH* activates the memetic part of CoMPSO. The local search operators, defined by the memes in M, are applied in a probabilistic way: m_i is applied to the personal best position of particle p_i with a probability γ at every ϕ iteration, and the local search application is denoted by $m_i(p_i)$ or equivalently by $x_i^{(m)}(b_i^{(p)})$. Furthermore, for every iteration, the local search is also applied to the global best particle in position $g_t^{(p)}$. In this case, the global best particle and its respective meme are indicated by p_g and m_g. Before being applied, the memes are evolved by means of the *PSO_MEME* evolution scheme described in Section 2.2.4. If the meme application leads to a fitness improvement, the new candidate solution obtained replaces both the personal best that the particle current position. Finally, the global best is updated accordingly.

Algorithm 1 CoMPSO Pseudo-Code.

procedure CoMPSO
  $t \leftarrow 0$
  $P \leftarrow$ INITIALIZE_PARTICLES()
  $f(P) \leftarrow$ EVALUATE_FITNESS()
  $M \leftarrow$ INITIALIZE_MEMES()
  while termination criterion is not met **do**
    $t \leftarrow t + 1$
    if DIV_CONTROL() **then**
      DIV_RESTORE()
    end if
    for all $p_i \in P$ **do**
      $v_{i,t}^{(p)} \leftarrow$ UPDATE_VELOCITY$(v_{i,t-1}^{(p)}, x_{i,t-1}^{(p)}, b_{i,t-1}^{(p)}, g_{t-1}^{(p)})$
      $x_{i,t}^{(p)} \leftarrow$ UPDATE_POSITION$(x_{i,t-1}^{(p)}, v_{i,t}^{(p)})$
      $f(x_{i,t}^{(p)}) \leftarrow$ EVALUATE_FITNESS$(x_{i,t}^{(p)})$
      $b_{i,t}^{(p)} \leftarrow$ UPDATE_PERSONAL_BEST$(b_{i,t-1}^{(p)}, x_{i,t}^{(p)})$
    end for
    $g_t^{(p)} \leftarrow$ UPDATE_GLOBAL_BEST()
    if LOCAL_SEARCH(γ, ϕ) **then**
      for all $p_i \in P$ **do**
        PSO_MEME(m_i)
        $m_i(p_i)$
      end for
    end if
    PSO_MEME(m_g)
    $m_g(p_g)$
    $g_t^{(p)} \leftarrow$ UPDATE_GLOBAL_BEST()
  end while
  return getBestSolution
end procedure

2.2.2. Diversity Control

A diversity control mechanism has been introduced in order to avoid stagnation or premature convergence to local optima. When premature convergence is detected, a subset of particles whose positions will be reinitialized is selected. The method consists of two main components: a *diversity measure* δ and a *diversity restore mechanism* invoked when the population diversity becomes too low according to measure δ.

The diversity measure is computed on the fitness values of the particle population according to

$$\delta(P_t) \leftarrow \text{std}(f(P_t)) \tag{3}$$

where $\text{std}(f(P_t))$ represents the standard deviation of the fitness values of particle population P at time t. It should be noted that, although genotypic distances between particles seem to be in principle more appropriate than fitness values, it has been experimentally observed that the use of

these latter indicators provides a good diversity measure with the additional property of a more efficient computation (this is due to the fact that the computation is performed on $\mathbb{R}$ and not on $\mathbb{R}^d$).

The diversity $\delta(P_0)$ of the initial random population P_0 is employed to compute the threshold value $\tau = 0.2 \cdot \delta(P_0)$ used in the routine *DIV_CONTROL* in order to recognize population convergence or stagnation. This threshold value is computed basing on the unbiased particle population at time 0 since it is the only one generated in a pure random way.

Therefore, at each iteration, in the case that $\delta(P_t) < \tau$, the *DIV_RESTORE* procedure is invoked. This routine randomly restarts the positions of the worst fit 50% of the population. Only the current positions of particles are restarted, while velocities and personal best values are kept unchanged.

2.2.3. Meme Representation

The local search operators adopted in CoMPSO are a generalization of the random walk (RW) technique [41–43]. RW is an iterative and stochastic local optimization method. Let y_t be the candidate (local) minimum at the t-th iteration. Then the new value y_{t+1} is computed according to

$$y_{t+1} \leftarrow \begin{cases} y_t + w_t z_t & \text{if } f(x_t + w_t z_t) < f(x_t) \\ y_t & \text{otherwise} \end{cases} \tag{4}$$

where w_t is a scalar step-length, and z_t is a unit-length random vector. The step length w_t is initialized to a given value w_0 and is halved at the end of every iteration in the case that the fitness has not been improved. The process is repeated until a given number q of iterations is reached.

In our generalization, other than the previously described parameters w_0 and q, two other parameters are introduced: the ramification factor b and the number of chosen successors k (with $k \leq b$). Briefly, the idea is to expand RW in a (b-k-bounded) breadth search conversely from the pure depth-first style of the original RW. Initially, k copies of the starting (or seed) point of RW are generated. Then b new points are generated from the current k ones following Rule (4). Therefore, from these b intermediate points, the k-fittest ones are chosen as next-iteration current points, and the process is iterated until the deepness parameter q is reached.

Using this local search scheme, a generic meme is represented by means of the following four parameters: a real value w_0 and the three integers b, k, and q. Furthermore, for each parameter, a range of admissible values has been experimentally established: $w_0 \in [0.5, 4]$, $b \in [1, 8] \cap \mathbb{N}$, $k \in [1, b] \cap \mathbb{N}$, and $q \in [4, 16] \cap \mathbb{N}$.

2.2.4. Meme Evolution

As introduced in the previous section, a meme is represented in the hybrid (continuous/discrete) meme space by the 4-ple $m = (w_0, b, k, q)$. Meme PSO evolution proceeds asynchronously with respect to particle evolution, this is due to the fact that memes are evolved only before they are applied and according to probability γ and frequency ϕ described above.

Memes, like particles, at each iteration t, are characterized by a position $x_{i,t}^{(m)}$, i.e., the representation of the meme in the meme space, a velocity $v_{i,t}^{(m)}$ (only for the continuous component), a personal best position $b_{i,t}^{(m)}$, and a global best meme $g_t^{(m)}$. Two alternative functions have been considered as meme fitness $f^{(m)}$: (1) the "absolute" fitness f of the particle p_i associated to meme m_i, and (2) the fitness improvement Δf realized with the application of the meme m_i to particle p_i. Some preliminary experiments have shown that the two approaches do not significantly differ.

Memes are evolved by a classical PSO scheme modified to manage the meme vectors composed by hybrid continuous/discrete components. The *PSO_MEME* scheme distinguishes the evolution of continuous components (according to Update Rules (1) and (2)) and that of discrete components where a probabilistic technique, presented in the next subsection, is applied. For the sake of completeness, *PSO_MEME* pseudo-code is reported in Algorithm 2.

Algorithm 2 Meme Evolution Pseudo-Code

procedure PSO_MEME(m_i)
  for each continuous dimension j **do**
    $v_{i,j}^{(m)} \leftarrow$ UPDATE_CONTINUOUS_VELOCITY($v_{i,j}^{(m)}, x_{i,j}^{(m)}, b_{i,j}^{(m)}, g_j^{(m)}$)
    $x_{i,j}^{(m)} \leftarrow$ UPDATE_CONTINUOUS_POSITION($x_{i,j}^{(m)}, v_{i,j}^{(m)}$)
  end for
  for each discrete dimension j **do**
    $x_{i,j}^{(m)} \leftarrow$ UPDATE_DISCRETE_POSITION($x_{i,j}^{(m)}, b_{i,j}^{(m)}, g_j^{(m)}$)
  end for
  $f^{(m)}(m_i) \leftarrow$ EVALUATE_MEME_FITNESS(m_i, p_i)
  $b_i^{(m)} \leftarrow$ UPDATE_MEME_PERSONAL_BEST($b_i^{(m)}, x_i^{(m)}$)
  $g^{(m)} \leftarrow$ UPDATE_MEME_GLOBAL_BEST($g^{(m)}, x_i^{(m)}$)
end procedure

2.2.5. Meme PSO for Discrete Domains

The meme discrete components are managed using a probabilistic technique that, by exploiting the total order of the integer domains here considered, simulates the classical PSO dynamics for continuous domains.

For each meme and for each discrete component domain $D_j = \{d_{j,1}, \ldots, d_{j,r}\}$, an appropriate probability distribution P_{D_j} over the values of D_j is built, then the new position x_j for component j is computed according to a randomized roulette wheel tournament among values in D_j and by using probabilities P_{D_j}.

As described in the following, the probability distribution is set in a way that simulates the properties of the classical continuous PSO [44].

Probability Distribution on Discrete Components Let $d_x, d_b, d_g \in D_j$ represents the values: the current meme position, the meme personal best position, and the global best meme position for component j of the meme encoding, then the distribution D_j is defined as a mixture of four discrete probability distributions:

1. a uniform distribution that assigns probability $1/|D_j|$ (where $|D_j|$ is the cardinality of D_j) to every value $d_i \in D_j$,
2. a triangular distribution centered in d_x,
3. a triangular distribution centered in d_b, and
4. a triangular distribution centered in d_g.

Each triangular distribution is determined by the center value d_k, an *amplification* factor $\alpha > 1$ and a *width* $2 \cdot \lambda + 1$. The probability of d_k is amplified by the factor α, i.e., the previous probability $P_{D_j}(d_k)$ is multiplied by α, while the $2 \cdot \lambda$ values of domain D_j located around d_k, i.e., $d_{k-\lambda}, \ldots, d_{k-1}$ and $d_{k+1}, \ldots, d_{k+\lambda}$, are amplified by a *smoothing* factor $\beta = \frac{\alpha}{\lambda+1} \cdot |\lambda + 1 - s|$, where s is the distance of the value from the center d_k. Moreover, it should be noted that the probabilities of each triangular distribution are normalized in order to sum up to 1.

The amplification factor α are set by using the PSO parameters; that is $1 + \omega$ for the center d_x, $1 + \varphi_1$ for d_b, and $1 + \varphi_2$ for d_g.

Discrete position update. The new position of a discrete component x_j is computed by a random roulette wheel tournament that uses the probability distribution above.

The technique for discrete components based on probability distribution values can be interpreted as initially considering all the values as equally probable, except for d_x, d_b, and d_g, whereas the

amplification factors based on classical PSO parameters ω, φ_1, φ_2 confer a greater probability. Moreover, the smoothness factors, βs, also amplify the probabilities of centers neighbors, i.e., values close to the centers.

In other words, amplification and smoothing factors implement, for the discrete components, the probabilistic counterparts of the typical behavior of PSO velocity, i.e., the tendency to remain in the current position (inertia) and the tendency to move toward a personal and global best (cognitive and social dynamics). It should be noted that a similar idea is present in [45].

3. Experiments

The performances of CoMPSO have been evaluated on the set of five benchmark functions reported in Table 1. Those functions differ from each other for the properties of modality, symmetry around the optimum, and regularity.

In each run of CoMPSO, termination conditions have been defined for convergence and maximum computational resources. An execution is regarded convergent if $f(x) - f(x^{opt}) < \epsilon$. On the other hand, the execution has been considered terminated unsuccessfully if the number of function evaluations (NFES) exceeds the allowed cap of 100,000. The dimensionalities, feasible ranges, and ϵ values, used for each benchmark, are also reported in Table 1.

Table 1. Benchmark functions.

f	Dim.	Range	ϵ
$f_1(x) = \sum_{i=1}^{d} x_i^2$	30	$[-100, 100]^{30}$	10^{-2}
$f_2(x) = \sum_{i=1}^{d} x_i^2/4000 - \prod_{i=1}^{d} \cos(x_i/\sqrt{i}) + 1$	30	$[-600, 600]^{30}$	10^{-1}
$f_3(x) = 0.5 + \left\{\sin^2(\sqrt{x_1^2 + x_2^2}) - 0.5\right\} / \left\{(1 + 0.001(x_1^2 + x_2^2))^2\right\}$	2	$[-100, 100]^2$	10^{-5}
$f_4(x) = -20\exp\left\{-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right\} - \exp\left\{\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right\} + 20 + e$	30	$[-32, 32]^{30}$	10^{-3}
$f_5(x) = \sum_{i=1}^{d} \begin{cases} 0.15 \cdot (z_i - 0.05\mathrm{sgn}(z_i)^2) \cdot h_i & \text{if } \lvert x_i - z_i\rvert \le 0.05 \\ h_i \cdot x_i^2 & \text{otherwise} \end{cases}$	4	$[-1000, 1000]^4$	10^{-7}

Following the setup of [20], three different swarm sizes have been considered, namely 15, 30, and 60. The other parameters were set by using the PSO standard values suggested in [46], i.e., $\omega = 0.7298$, $\varphi_1 = \varphi_2 = 1.49618$. The domain ranges defining the meme space were $b, k \in [1, 8] \cap \mathbb{N}$, $q \in [1, 16] \cap \mathbb{N}$, and $w \in [0.5, 4]$, while an amplification width $\lambda = 4$ was used for the evolution of the memes' discrete features. Furthermore, meme application probability and frequency were set to $\gamma = 0.2$ and $\phi = 5$.

For each swarm size configuration, a series of 50 executions was held in order to generate more confidence in the statistical results. In each execution series, the success rate SR (i.e., the number of convergent executions above the total number of executions), the average NFES of all convergent executions C, and the quality measure $Q_m = C_{avg}/SR$ introduced in [47] are recorded.

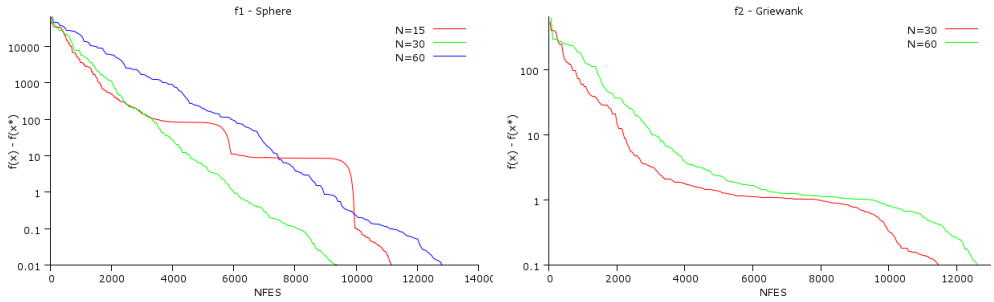
All these previously described performance indexes are reported in Table 2 for CoMPSO, classical PSO (CPSO), and static memetic PSO (SMPSO), i.e., PSO endowed with an RW local search operator but without meme evolution [20], which is the only comparable memetic PSO algorithm in the literature. The best quality measure in every line of Table 2 is provided in bold.

Table 2. Experimental results.

f	n	CoMPSO			CPSO			SMPSO		
		SR	C	Q_m	SR	C	Q_m	SR	C	Q_m
f_1	15	1.00	14,324	**14,324**	0.00	-	-	0.00	-	-
	30	1.00	12,226	12,226	1.00	11,235	**11,235**	0.50	37,409	74,818
	60	1.00	14,254	14,254	1.00	12,680	**12,680**	1.00	19,395	19,395
f_2	15	0.00	-	-	0.00	-	-	0.00	-	-
	30	0.96	12,306	**12,819**	0.50	8655	17,310	0.58	12,807	22,081
	60	1.00	15,497	15,497	1.00	11,480	**11,480**	0.70	15,785	22,550
f_3	15	1.00	14,697	14,697	0.28	5475	19,553	0.83	6671	**8037**
	30	1.00	24,433	24,433	0.64	10,935	**17,086**	1.00	27,247	27,247
	60	1.00	15,672	**15,672**	0.70	29,580	42,257	1.00	22,993	22,993
f_4	15	1.00	45,184	**45,184**	0.00	-	-	1.00	57,385	57,385
	30	1.00	41,077	**41,077**	0.00	-	-	1.00	43,673	43,673
	60	1.00	47,610	47,610	0.06	49,350	822,500	1.00	38,137	**38,137**
f_5	15	1.00	2348	2348	1.00	1642	**1642**	0.98	2660	2714
	30	1.00	3279	3279	1.00	2560	**2560**	1.00	3938	3938
	60	1.00	5206	5206	1.00	4220	**4220**	1.00	5285	5285

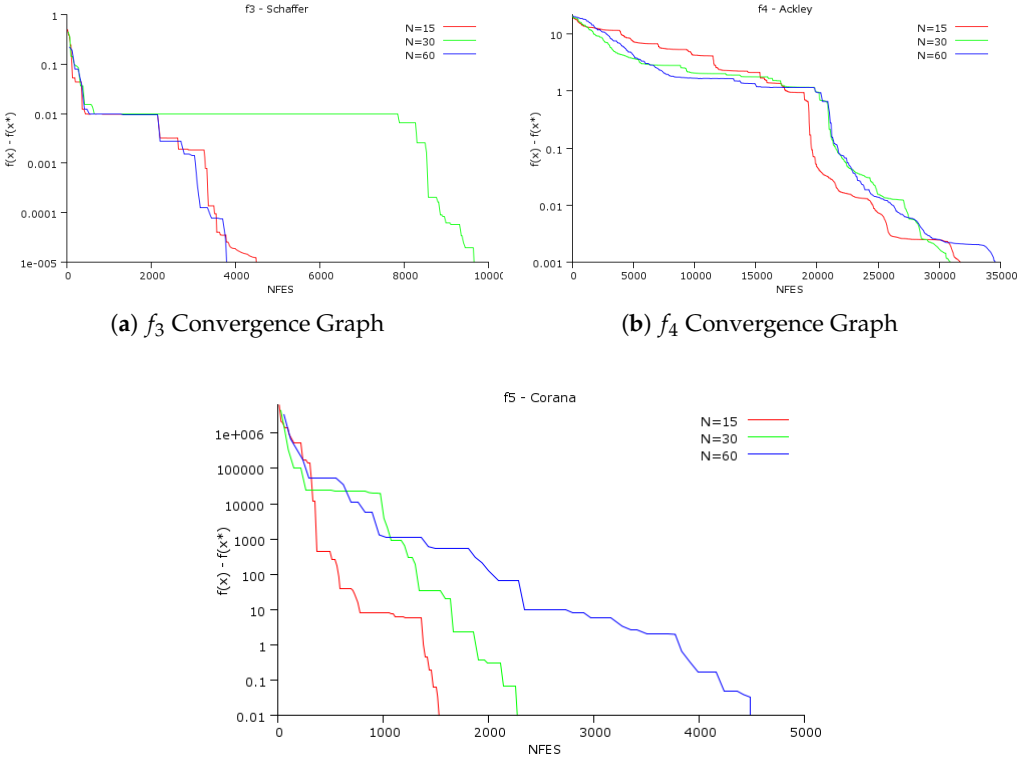
These results clearly show that the CoMPSO approach greatly improves the success rate of CPSO and SMPSO. In particular, it must be noted that CoMPSO converges almost everywhere, and it has a remarkable worst case convergence probability of 96%. On the other hand, CPSO, in some cases, fails to converge at all when a small swarm size is employed. Finally, the CoMPSO convergence speed is comparable to that of the SMPSO, although, as expected, in the more simple cases, i.e., f_1 and f_5, the convergence speed of CoMPSO is worse than that of CPSO, which is likely due to the NFES overhead introduced by local search operators.

Figures 1a,b and 2a–c plot the CoMPSO convergence graph with respect to the different swarm sizes adopted. These graphs show that CoMPSO appears quite monotonic with respect to swarm size and that a low number of particles seems to be generally preferable.



(**a**) f_1 Convergence Graph (**b**) f_2 Convergence Graph

Figure 1. (**a**,**b**) CoMPSO convergence graphs for f_1 and f_2.



(**a**) f_3 Convergence Graph

(**b**) f_4 Convergence Graph

(**c**) f_5 Convergence Graph

Figure 2. (**a**–**c**) CoMPSO convergence graphs for f_3, f_4, and f_5.

Finally, a measure of meme convergence is shown in Figure 3, which shows the evolution of meme standard deviation (meme STD) on benchmark f_4. Meme convergence is fast in the early stage and, as expected, remains fairly constant, with only small adaptations, during the rest of the computation. The meme convergence curve, together with the quality measures and the success rates, show the effectiveness of CoMPSO and its ability to adapt its local refinement behavior to the landscape of the problem at hand.

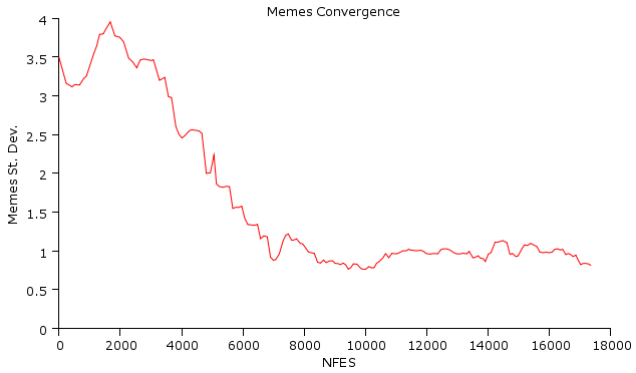


Figure 3. Memes Convergence Graph.

4. Discussion

In this paper, a coevolving memetic PSO (CoMPSO), characterized by two co-evolving populations of particles and memes, has been presented. The main contribution of this work is represented by the meme evolution technique that allows one to enhance the effectiveness of the PSO approach. Memetic algorithms (MAs) have recently received great attention as effective meta-heuristics to

improve general evolutionary algorithm (EA) schemes by combining EAs with local search procedures and have demonstrated to be a very effective method for performance improvement. However, to the best of our knowledge, this is the first work where a memetic PSO algorithm with meme co-evolution has been proposed. Since memes have been described using one real and three integer parameters, a probabilistic PSO evolution technique for the discrete components in the meme representation has been designed. This technique is inspired from our previous work [45] and preserves typical PSO behavior of cognitive, social, and momentum dynamics.

The algorithm has been tested on some standard benchmark problems, and the results presented here show that CoMPSO outperforms the success rates of both classical PSO and static memetic PSO [20], although its convergence speed is affected by the overhead due to the local search applications. The effectiveness of the method relies on the ability to dynamically adapt the local search operators, i.e., the memes, to the problem landscape at hand. While experiments have been held on a limited set of standard benchmarks, the goal of demonstrating the feasibility of the approach i.e., providing the adaptivity of memes during searches in PSO, and improving the previous results can be considered achieved.

Future and ongoing works have been designed on different perspectives: from an experimental point of view, we are currently holding systematic experiments on larger sets of benchmarks, and we are planning experiments with hybrid continuous/discrete problems [48]; from the theoretical model point of view, we are investigating different models and synchronization mechanisms for the meme operators, we are also currently designing a self-regulatory mechanism for the CoMPSO parameters and investigating its applications to different classes of problems, such as multiobjective and multimodal problems [49].

Author Contributions: The authors equally contributed to this work, thus they share first-author contribution.

Funding: The research described in this work has been partially supported by: the research grant "Fondi per i progetti di ricerca scientifica di Ateneo 2019" of the University for Foreigners of Perugia under the project "Algoritmi evolutivi per problemi di ottimizzazione e modelli di apprendimento automatico con applicazioni al Natural Language Processing".

Acknowledgments: We would like to thank the reviewers for their valuable comments.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Neri, F.; Cotta, C. Memetic algorithms and memetic computing optimization: A literature review. *Swarm Evol. Comput.* **2012**, *2*, 1–14. [CrossRef]
2. Akhshabi, M.; Tavakkoli-Moghaddam, R.; Rahnamay-Roodposhti, F. A hybrid particle swarm optimization algorithm for a no-wait flow shop scheduling problem with the total flow time. *Int. J. Adv. Manuf. Technol.* **2014**, *70*, 1181–1188. [CrossRef]
3. Wu, X.; Che, A. A memetic differential evolution algorithm for energy-efficient parallel machine scheduling. *Omega* **2019**, *82*, 155–165. [CrossRef]
4. Liu, B.; Wang, L.; Jin, Y. An effective PSO-based memetic algorithm for flow shop scheduling. *IEEE Trans. Syst. Man Cybern.* **2007**, *37*, 18–27. [CrossRef]
5. Huang, H.; Qin, H.; Hao, Z.; Lim, A. Example-based Learning Particle Swarm Optimization for Continuous Optimization. *Inf. Sci.* **2012**, *182*, 125–138. [CrossRef]
6. Tang, J.; Lim, M.; Ong, Y. Diversity-adaptive parallel memetic algorithm for solving large scale combinatorial optimization problems. *Soft Comput.* **2007**, *11*, 873–888. [CrossRef]
7. Beheshti, Z.; Shamsuddin, S.M.; Hasan, S. Memetic binary particle swarm optimization for discrete optimization problems. *Inf. Sci.* **2015**, *299*, 58–84. [CrossRef]
8. Li, X.; Wei, J.; Liu, Y. A Memetic Particle Swarm Optimization Algorithm to Solve Multi-objective Optimization Problems. In Proceedings of the 2017 13th International Conference on Computational Intelligence and Security (CIS), Hong Kong, China, 15–18 December 2017; pp. 44–48. [CrossRef]

9. Liu, D.; Tan, K.C.; Goh, C.; Ho, W.K. A Multiobjective Memetic Algorithm Based on Particle Swarm Optimization. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2007**, *37*, 42–50. [CrossRef]
10. Cao, Y.; Zhang, H.; Li, W.; Zhou, M.; Zhang, Y.; Chaovalitwongse, W.A. Comprehensive Learning Particle Swarm Optimization Algorithm with Local Search for Multimodal Functions. *IEEE Trans. Evol. Comput.* **2018**. [CrossRef]
11. Wang, H.; Moon, I.; Yang, S.; Wang, D. A Memetic Particle Swarm Optimization Algorithm for Multimodal Optimization Problems. *Inf. Sci.* **2012**, *197*, 38–52. [CrossRef]
12. Sengupta, S.; Basak, S.; Peters, R.A. Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives. *Mach. Learn. Knowl. Extr.* **2018**, *1*, 157–191. [CrossRef]
13. Hart, W.; Krasnogor, N.; Smith, J. *Recent Advances in Memetic Algorithms*; Springer: Berlin/Heidelberg, Germany, 2005; Volume 166.
14. Cotta, C.; Mathieson, L.; Moscato, P., Memetic Algorithms. In *Handbook of Heuristics*; Martí, R., Pardalos, P.M., Resende, M.G.C., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 607–638. [CrossRef]
15. Gupta, A.; Ong, Y.S. Canonical Memetic Algorithms. In *Memetic Computation: The Mainspring of Knowledge Transfer in a Data-Driven Optimization Era*; Springer International Publishing: Cham, Switzerland, 2019; pp. 17–26. [CrossRef]
16. Kirkpatrick, S.; Gelatt, C.; Vecchi, M. Optimization by simulated annealing. *Science* **1983**, *220*, 671. [CrossRef]
17. Michalewicz, Z. *Genetic Algorithms + Data Structures = Evolution Programs*; Springer: Berlin, Germany, 1992; Volume 19.
18. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the 1995 IEEE International Conference on Neural Networks, Perth, Western Australia, 27 November–1 December 1995; Volume 4, pp. 1942–1948.
19. Eusuff, M.; Lansey, K.; Pasha, F. Shuffled frog-leaping algorithm: A memetic meta-heuristic for discrete optimization. *Eng. Optim.* **2006**, *38*, 129–154. [CrossRef]
20. Petalas, Y.; Parsopoulos, K.; Vrahatis, M. Memetic particle swarm optimization. *Ann. Oper. Res.* **2007**, *156*, 99–127. [CrossRef]
21. Ong, Y.; Keane, A. Meta-Lamarckian learning in memetic algorithms. *IEEE Trans. Evol. Comput.* **2004**, *8*, 99–110. [CrossRef]
22. Moscato, P. On evolution, search, optimization, genetic algorithms and martial arts: Towards memetic algorithms. In *Caltech Concurrent Computation Program, C3P Rep.*; California Institute of Technology: Pasadena, CA, USA, 1989; Volume 826, p. 1989.
23. Dawkins, R. *The Selfish Gene*; Oxford University Press: New York, NY, USA, 2006.
24. Carpi, A.; de Luca, A. Central Sturmian Words: Recent Developments. In *Developments in Language Theory 2005*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; Volume 3572, pp. 36–56.
25. Smith, J. Coevolving memetic algorithms: A review and progress report. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2007**, *37*, 6–17. [CrossRef]
26. Baioletti, M.; Milani, A.; Santucci, V. A New Precedence-Based Ant Colony Optimization for Permutation Problems. In *Simulated Evolution and Learning*; Springer International Publishing: Cham, Switzerland, 2017; pp. 960–971. [CrossRef]
27. Baioletti, M.; Milani, A.; Santucci, V. Algebraic Particle Swarm Optimization for the permutations search space. In Proceedings of the 2017 IEEE Congress on Evolutionary Computation, CEC 2017, Donostia, San Sebastián, Spain, 5–8 June 2017; pp. 1587–1594. [CrossRef]
28. Ahmed, A.; Sun, J. Bilayer Local Search Enhanced Particle Swarm Optimization for the Capacitated Vehicle Routing Problem. *Algorithms* **2018**, *11*, 31. [CrossRef]
29. Yu, X.; Estevez, C. Adaptive Multiswarm Comprehensive Learning Particle Swarm Optimization. *Information* **2018**, *9*, 173. [CrossRef]
30. Baioletti, M.; Milani, A.; Santucci, V. Automatic Algebraic Evolutionary Algorithms. In Proceedings of the International Workshop on Artificial Life and Evolutionary Computation (WIVACE 2017), Venice, Italy, 19–21 September 2017; pp. 271–283. [CrossRef]

31. Santucci, V.; Baioletti, M.; Milani, A. A Differential Evolution Algorithm for the Permutation Flowshop Scheduling Problem with Total Flow Time Criterion. In Proceedings of the 13th International Conference on Parallel Problem Solving from Nature—PPSN XIII, Ljubljana, Slovenia, 13–17 September 2014; pp. 161–170. [CrossRef]
32. Dai, H.; Chen, D.; Zheng, Z. Effects of Random Values for Particle Swarm Optimization Algorithm. *Algorithms* **2018**, *11*, 23. [CrossRef]
33. Letting, L.; Hamam, Y.; Abu-Mahfouz, A. Estimation of Water Demand in Water Distribution Systems Using Particle Swarm Optimization. *Water* **2017**, *9*, 593. [CrossRef]
34. Akarsh, S.; Kishor, A.; Niyogi, R.; Milani, A.; Mengoni, P. Social Cooperation in Autonomous Agents to Avoid the Tragedy of the Commons. *Int. J. Agric. Environ. Inf. Syst.* **2017**, *8*, 1–19. [CrossRef]
35. Oldewage, E.T.; Engelbrecht, A.P.; Cleghorn, C.W. Boundary Constraint Handling Techniques for Particle Swarm Optimization in High Dimensional Problem Spaces. In *Swarm Intelligence*; Dorigo, M., Birattari, M., Blum, C., Christensen, A.L., Reina, A., Trianni, V., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 333–341.
36. De Luca, A.; Carpi, A. Uniform words. *Adv. Appl. Math.* **2004**, *32*, 485–522. [CrossRef]
37. Hamed, H.N.A.; Kasabov, N.; Michlovský, Z.; Shamsuddin, S.M.H. String Pattern Recognition Using Evolving Spiking Neural Networks and Quantum Inspired Particle Swarm Optimization. In Proceedings of the 16th International Conference on Neural Information Processing, ICONIP 2009, Bangkok, Thailand, 1–5 December 2009; pp. 611–619. [CrossRef]
38. D'Alessandro, F.; Carpi, A. Independent sets of words and the synchronization problem. *Adv. Appl. Math.* **2013**, *50*, 339–355. [CrossRef]
39. De Luca, A.; Carpi, A. Harmonic and gold Sturmian words. *Eur. J. Comb.* **2004**, *25*, 685–705. [CrossRef]
40. Carpi, A. On the repetition threshold for large alphabets. In *Mathematical Foundations of Computer Science 2006*; Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2006; Volume 4162, pp. 226–237.
41. Bordenave, C.; Caputo, P.; Salez, J. Random walk on sparse random digraphs. *Probab. Theory Relat. Fields* **2018**, *170*, 933–960. [CrossRef]
42. Weiss, G. *Aspects and Applications of the Random Walk (Random Materials & Processes S.)*; North-Holland: New York, NY, USA, 1994.
43. Mengoni, P.; Milani, A.; Li, Y. Clustering students interactions in eLearning systems for group elicitation. In *Computational Science and Its Applications—ICCSA 2018*; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2018; Volume 10962, pp. 398–413. [CrossRef]
44. Baioletti, M.; Milani, A.; Santucci, V. Learning Bayesian Networks with Algebraic Differential Evolution. In *Parallel Problem Solving from Nature—PPSN XV*; Springer International Publishing: Cham, Switzerland, 2018; pp. 436–448. [CrossRef]
45. Santucci, V.; Milani, A. Particle Swarm Optimization in the EDAs Framework. In *Soft Computing in Industrial Applications*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 87–96. [CrossRef]
46. Clerc, M.; Kennedy, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
47. Feoktistov, V. *Differential Evolution: In Search of Solutions*; Springer: New York, NY, USA, 2006; Volume 5.
48. Baioletti, M.; Milani, A.; Santucci, V. An Extension of Algebraic Differential Evolution for the Linear Ordering Problem with Cumulative Costs. In *Parallel Problem Solving from Nature—PPSN XIV*; Springer: Cham, Switzerland, 2016; pp. 123–133. [CrossRef]
49. Baioletti, M.; Milani, A.; Santucci, V. MOEA/DEP: An Algebraic Decomposition-Based Evolutionary Algorithm for the Multiobjective Permutation Flowshop Scheduling Problem. In *Evolutionary Computation in Combinatorial Optimization—EvoCOP 2018*; Springer International Publishing: Cham, Switzerland, 2018; pp. 132–145. [CrossRef]