Advances in Engineering Software 90 (2015) 127–137

Contents lists available at ScienceDirect

Advances in Engineering Software

journal homepage: www.elsevier.com/locate/advengsoft

A new hybrid PSO algorithm based on a stochastic Markov chain model

N. Di Cesare*, D. Chamoret, M. Domaszewski

Université Bourgogne Franche-Comté, UTBM, IRTES-EA 7274, M3M, France

ARTICLE INFO

Article history:
Received 24 April 2015
Revised 23 June 2015
Accepted 18 August 2015

Keywords:
Particle Swarm Optimization
PageRank
Optimization
Markov chains
Population topology
Algorithms

ABSTRACT

Based on the recent research concerning the PageRank Algorithm used in the famous search engine Google [1], a new Inverse-PageRank-Particle Swarm Optimizer (I-PR-PSO) is presented in order to improve the performances of classic PSO. The resulted algorithm uses a stochastic Markov chain model to define an intelligent topological structure of the swarm's population, in which the better particles have an important influence on the others. In the presented experiments, calculations on some benchmark functions classically used to test optimization methods are performed, and the results are compared to different versions of the standard PSO, that is using different topological structures of the population. The experimental results show that I-PR-PSO can converge quicker on the tested functions, and can find better results in the solution domain than its tested peers.

1. Introduction, state of the art

For decades, the field of optimization has been explored as an active research area. An unconstrained optimization problem can be formulated as D-dimensional minimization problems as follow

$$Min\, f(\vec{x}) \qquad \vec{x} = (x_1, x_2, x_3, \ldots, x_D) \tag{1}$$

where D is the number of design variables to be optimized, that is the dimension of the problem, and f is the objective function to minimize. The past few years saw the development of many different optimization techniques. The population-based metaheuristic methods have been demonstrated and defined as very useful and efficient, even though there is not any mathematical evidence of their convergence to the global optimum. In fact, those methods consider a population of solutions instead of a single one. Using some stochastic parameters, they can converge efficiently to the global optimum. They are generally inspired by physical or biological phenomena, such as the Ant Colony Optimization [2] which draws its inspiration from the foraging behavior of some ant species, the evolutionary algorithms [3–5] which mimic the process of natural evolution, using processes such as inheritance, mutation, selection and crossover. The Particle Swarm Optimization Algorithm (PSO) comes from the observation of some flocks of birds by Reynolds [6] in 1987, and has been developed by Kennedy and Eberhart [7] in 1995. Understanding how the birds can achieve their complex and optimal movement, a new optimization method which uses a swarm of potential solutions has been proposed. Because of stochastic parameters, these solutions can follow the best particles and converge together to the global optimum of the considered objective function. Lately, lots of different improvements of PSO concerning the population's topology have been presented in the literature [8–18]. The population's topology defines how the particles are structured, thus defines the influence they have on each others. The first population's topology proposed in the literature is statical. Therefore, each particle is always influenced by the same other particles all along the calculation. For example, Eberhart and Kennedy [9] have developed the well known LBEST[1] and GBEST[2] topologies. In the classical GBEST population topology, the entire population is treated as the individual's neighborhood [19]. Eventually, the particles are influenced by the global best one, as one can see in Fig. 1. In the local LBEST version, the particles are linked with two of the other particles. The population topology is then a ring, as one can see in Fig. 2, and the best performance of each particular neighborhood is chosen between the two particles of its neighborhood.

In their work, Mendes et al. [19] proposed different statical population topologies, such as the pyramid, which is a three-dimensional wire-frame triangle, the Von Neumann, which is a square lattice whose extremities connect as a torus, and the four clusters one, in which four clusters of particles are completely interconnected, connected among themselves by a few short-cuts, as one can see in Fig. 3. An approach in which the quality of the solution is considered in a weighted definition of the particles' moving has also been presented in [8]. In fact, it has been noted by Mendes et al. [20] that all the neighbors of a particle can be a source of influence.

* Corresponding author. Tel.: +33 3 84 58 39 17.
E-mail address: noelie.di-cesare@utbm.fr (N. Di Cesare).

[1] Local Best particle's topology
[2] Global Best particle's topology

http://dx.doi.org/10.1016/j.advengsoft.2015.08.005
0965-9978/© 2015 Elsevier Ltd. All rights reserved.

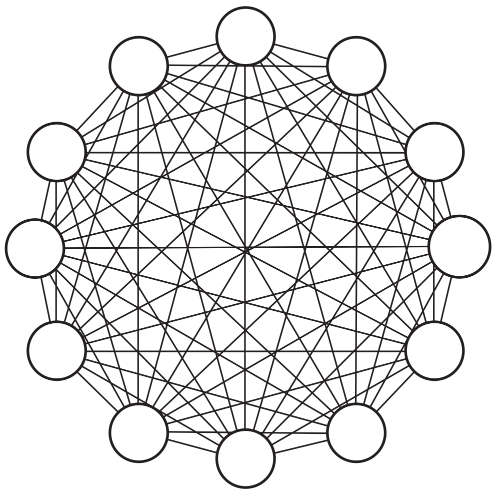
Fig. 1. The famous GBEST population topology.

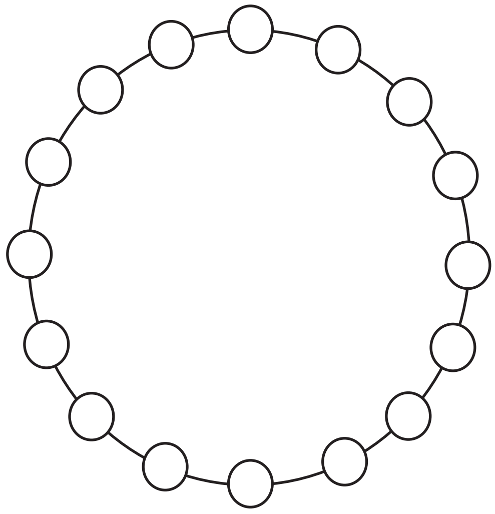
Fig. 2. The famous LBEST population topology.

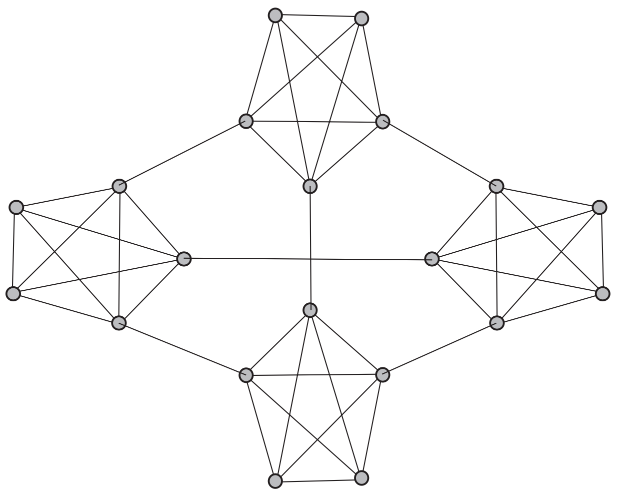
Fig. 3. The 4-clusters population topology.

The second type of population topology is the evolutive one. This type defines a population topology which is able to change through the iterations of the optimization algorithm. Akbari and Ziarati have applied the concept of ranking to the Particle Swarm Optimization [8]. In this work, at each iteration, the particles are sorted on the basis of their fitness value. Then, the γ best particles are used to influence the moving of the other particles. γ decreases during the iterations, thus the particles are less linked to the others during the optimization process: this algorithm starts with a GBEST topology, and finally the particles are only influenced by the global best one. This type of evolutionary topology has been also used in the work of Suganthan [13] in which the swarm starts linked as a LBEST topology. Then the number of links between the particles are extended during the PSO iterations, to finish with a GBEST topology. In a simpler way, Pasupuleti and Battiti [14] proposed to use only the best particle of the swarm to influence the others in his Gregarious Particle Swarm Optimizer. Janson and Middendorf [15] suggested a hierarchical Particle Swarm Optimizer in which all particles are arranged in a hierarchy tree that defines the neighborhood structure. The particle which achieves the global best fitness is the tree's root. If a particle finds a better solution than the one found by its direct hierarchical superior in the tree, the particles switch their places. Then, an evolutive topology of the population is provided, and the results proposed are globally better than the classical versions of PSO, that is with the GBEST and LBEST topologies. Jiang et al. [10], Lovbjerg et al. [16] and Blackwell and Branke [18] have proposed to partition the population into sub-swarms to improve the ability of exploration and exploitation. Angeline [11] have proposed a selective mechanism which ranks the particles as a function of the obtained fitness. Then, the worst half part of the swarm is teleported in the area of the best half part, but keep in memory its own best performances. Then, the moving of the swarm can be compared to the evolutionary algorithms, because of the sudden teleportation of the particles in the solution domain. In their work, Mohais et al. [12] generated a random oriented graph, defining the influences of the particles on the others. The topology of the swarm can be redefined randomly, with the static probability p_r defined at the beginning of the calculation. In conclusion of this article, it has been shown that evolutionary topology can exhibit better results than algorithms using a statical topology.

In this paper, a new efficient population topology based on a stochastic Markov chain model, used as in the inverse PageRank ®algorithm, is proposed. The population topology has the ability to evolve, and the calculations of the particles' motions are smartly weighted considering the quality of the solution. The linked particles are then considered as a Markov chain, and the quality of the solutions defines the probability transition of the chain, which determines the influence of the particles on the others. Section 2 gives the mathematical bases concerning the classical PSO. In Section 3, the mathematical background concerning the Markov chains and the PageRank algorithm, as well as the inverse PageRank methodology are depicted. Then, based upon the previous mathematical theory, the newly developed I-PR-PSO is proposed. Section 4 describes the simulations performed to test and validate the new optimization process, and the obtained results. Finally, Section 5 concludes the work.

2. Review of standard PSO

Such as the Genetic Algorithms [21], or the Ant Colony Optimization [2], the Particle Swarm Algorithm (PSO) [7] is a population-based metaheuristic optimization method. In PSO, the potential solutions of the optimization problem, called particles, move in the solution domain with a velocity[3], which is adjusted as a function of the position of other particles. All the particles follow the best one during the iterations and converge together to the global optimum of the considered objective function. Then, in the linear version of PSO which considers

[3] Called "the velocity" in the literature, this parameter is actually the displacement of the particle in the solution domain.

the neighborhood of the particles, the position of a particle $n^o i$ at iteration $t+1$, noted $\boldsymbol{X}_i^{t+1}$ is given as a function of

- $\boldsymbol{X}_i^t$: The position of the particle i at the iteration t,
- $\boldsymbol{V}_i^t$: The velocity of the particle i at the iteration t,
- $\boldsymbol{G}_{i,best}^t$: The position of the best particle in the neighborhood of the particle, at iteration t,
- $\boldsymbol{P}_{i,best}^t$: The position of the best personal performance of the particle found at the iteration t.

The position change of each particle of the swarm is given in the following manner [22]

$$\begin{cases} \boldsymbol{V}_i^{t+1} = \omega \times \boldsymbol{V}_i^t + c_1 \times rand_1 \times (\boldsymbol{P}_{i,best}^{t+1} - \boldsymbol{X}_i^t) \\ \quad + c_2 \times rand_2 \times (\boldsymbol{G}_{i,best}^{t+1} - \boldsymbol{X}_i^t) \\ \boldsymbol{X}_i^{t+1} = \boldsymbol{X}_i^t + \boldsymbol{V}_i^{t+1} \end{cases} \tag{2}$$

where c_1 and c_2 are acceleration factors, ω is the inertia weight defined to control the influence of the previous velocity on the next one [23], and $rand_1$ and $rand_2$ are some random real numbers distributed in [0, 1].

The speed of the particles has to be constrained for the calculation to converge. The speed of the particles in then defined in $[-V_{max}; V_{max}]$ where V_{max} depends on the solution domain, such as $V_{max} = X_{max}$ where X_{max} is the maximum position of the particles in the domain.

3. A new hybrid PSO based on a stochastic Markov chain model

3.1. Mathematical background concerning the Markov chains

A discrete-time Markov chain is a mathematical system that describes the transitions from one state to another, both given in a state space. This stochastic mathematical process is characterized as memoryless, which means that the future and the past are independent from the present state. Formally, a Markov chain is a sequence of X_n random variables in a state space E, where X_n is the state of the process at discrete time n. Then, the Markov process is defined such as follow.

$$\forall\, n \geq 0, \forall\, (p_0, p_1, \dots, p_{n-1}, k, l) \in E^{n+2}, \text{we have}$$
$$P(X_{n+1} = l | X_n = k, X_{n-1} = p_{n-1}, \dots, X_0 = p_0) = P(X_{n+1} = l | X_n = k) \tag{3}$$

Markov chains can also be described by a sequence of oriented graphs, in which the edges of graph are weighted by the probabilities of going from one state at time n to the following state at time $n+1$. The process can then be written in a simpler way using the transition probability matrix. If the probability of moving from state k to state l in one time step is noted $P(l|k) = C_{k,l}$, then the stochastic transition matrix, also called the stochastic connectivity matrix of the Markov chain, is given by $C_{k,l}$, where k is the row number, and l the column number. Since the probability of transitioning from state k to the others is 1, this matrix is a right stochastic matrix and we have

$$\sum_l C_{k,l} = 1 \tag{4}$$

Generally speaking, the probability transition of going from one state to another one in m discrete time steps is given by $\boldsymbol{C}^m$. Thus, a stationary probability vector $\boldsymbol{\pi}$ is defined as the steady state of the Markov chain model and does not change under application of the transition matrix $\boldsymbol{C}$ over the iterations. $\boldsymbol{\pi}$ is thus defined by a left eigenvector of the probability matrix associated with eigenvalue 1, and we have

$$\boldsymbol{\pi}\boldsymbol{C} = \boldsymbol{\pi} \tag{5}$$

For a matrix with strictly positive entries, which is the case for the matrix $\boldsymbol{C}$ of a Markov chain, this vector is unique, and can be computed by observing that

$$\forall k \quad lim_{m\to\infty}(\boldsymbol{C}^m)_{k,l} = \boldsymbol{\pi}_l \tag{6}$$

Named after Larry Page, one of the founders of Google®, the PageRank algorithm is a powerfull method to rank the web pages. Actually, a page is important if it is pointed to by other important pages [24]. The web is then considered as an oriented graph, in which the nodes represent the webpages, and the links are weighted by the probability to click on. Thus, in the PageRank model, the web is considered as a Markov chain. The PageRank algorithm is detailed in Appendix.

3.2. Analogy with our topological structure of the PSO population

In PSO, the population of the swarm can be seen as an oriented graph. The nodes represent the particles, and the transition probabilities can be seen as the influences of the particles on the others. In this paper, a new PSO algorithm based on the inverse PageRank algorithm is proposed. In the PageRank algorithm, the stochastic connectivity matrix between the nodes of the graph is known, and the PageRank vector is searched. In this work, the exact opposite is done. As said by Newton, talking about the work presented in [25]: "Basically, we are doing the inverse of what Google does. They know the transition probabilities and compute the steady-state, we know the steady-state and compute the transition probabilities."[4] While the basic calculation of Google is presented in Fig. 4, the calculation proposed in this paper, that is an inverse PageRank calculation, is presented in Fig. 5.

The using of the connectivity matrix $\boldsymbol{C}$ in inverse Markov chains calculations has already been studied in the literature. The solution to this linear inverse problem in not unique, and has been addressed in the works of Gzyl and Velásquez [26,27] and Csiszar [28]. In those papers, the solution to this constrained linear inverse problem is obtained by identifying the transition matrix that satisfies a certain maximum entropy condition, satisfying a least-squares condition.

In Inverse-Page-Rank PSO (I-PR-PSO), to define the PageRank vector, that is the steady-state of the Markov chain, the relative success of each particle of the swarm is used. Then, at each iteration, the relative success of each particle k regarding the best one G_{best} is calculated as given in Eq. (7). The vector containing all the relative successes respectively to each swarm's particle is then normalized as given in Eq. (8).

$$\boldsymbol{\pi}_{target}^T(1,k) = \left| \frac{fitness(G_{best}) \times 100}{fitness(G_{best}) - fitness(P_k) + \epsilon} \right| \forall k \in [1,n] \tag{7}$$

where $fitness(\boldsymbol{X})$ represents the value of the objective function for the particle $\boldsymbol{X}$, and n is the number of particles in the swarm. The parameter ϵ (10^{-7} or 10^{-15} depending on the precision of the computer) is used in order to avoid a division by zero when $fitness(G_{best}) = fitness(P_k)$. Eq. (7) represents a classification of the particles based not on their ranks in the population but with respect to the distance from the global best particles G_{best}. If a particle P_k is close to the G_{best}, its value in $\boldsymbol{\pi}_{target}$ is big.

The advantage of the newly developed I-PR-PSO algorithm is to take into account not only the fitness of the individuals at current position but especially the history of the iterations. The memory of

[4] http://www.slate.com/blogs/future_tense/2013/03/27/google_pagerank_algorithm_markov_chains_and_cancer.html Last connexion: June, 30, 2014.

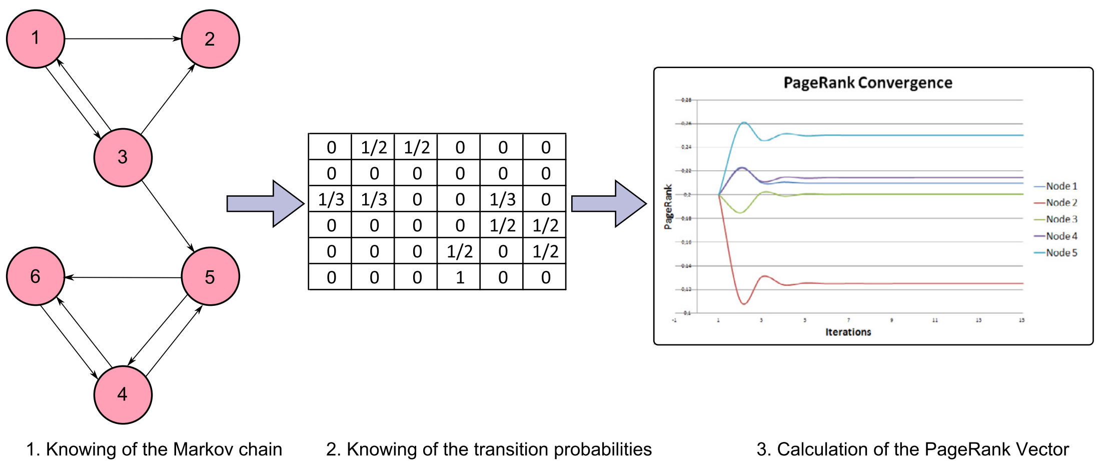

Fig. 4. The classic PageRank calculation done by the search engine developed by Google ®.

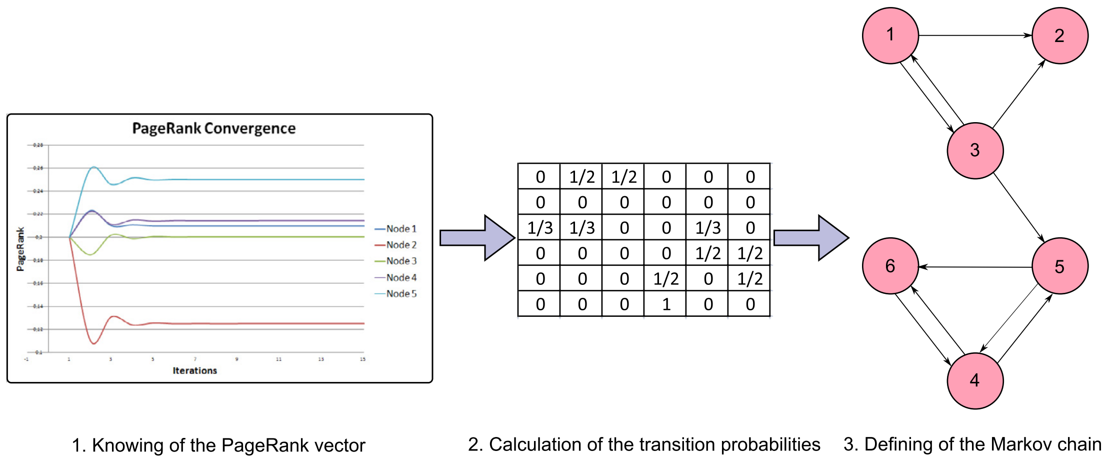

Fig. 5. The inverse calculation of a Markov chain model.

previous iterations is stored and used in the G_{best} variable in order to avoid premature convergence.

$$\boldsymbol{\pi}_{target}^T(1,k) \leftarrow \frac{\boldsymbol{\pi}_{target}^T(1,k)}{\sum_{k=1}^{D}(\boldsymbol{\pi}_{target}^T(1,k))} \quad \forall k \in [1,n] \tag{8}$$

where k is the k^{th} component of the vector $\boldsymbol{\pi}_{target}^T$.

As $\boldsymbol{\pi}_{target}^T$ is a probability vector, it is then normalized so that $\boldsymbol{\pi}_{target}^T(1,k) \in [0,1] \forall k$ and the sum of all its components is 1. This mathematical expression is effective only in the case of a minimization optimization problem. The vector defined by Eq. (7) is then considered as the target vector of the connectivity calculation. In fact, as it can be seen in Fig. 4, the PageRank vector is calculated knowing the stochastic connectivity matrix $\boldsymbol{C}$. Then, the purpose of our inverse PageRank algorithm is to find the stochastic connectivity matrix $\boldsymbol{C}$ (also called the "help matrix" in [29]) which fits with the previously defined target vector $\boldsymbol{\pi}_{target}^T$ (also called the "reputation vector" in [29]).

In I-PR-PSO, this target vector $\boldsymbol{\pi}_{target}^T$ defines the influence of each particle in the swarm according to their personal fitness. $\boldsymbol{\pi}_{target}^T$ can be seen as the steady state of the Markov chain defined by the graph of the PSO population topology. Its dimensions are $(1 \times n)$, where n is the number of nodes in the considered graph, that is the number of particles in the swarm. It can be seen that the sum of all of its components is equal to 1. Then, the goal of this work is to find the $(n \times n)$ connectivity matrix $\boldsymbol{C}$ defining the transition probabilities between the nodes of the considered graph, that is the influence of all the particles on the others, corresponding to this target vector. The constraints are $0 \leq C_{kl} \leq 1$ and $\sum_{l=1}^{n} C_{kl} = 1$.

In this way, the best particles will be the most influent among the swarm, and the worst ones will not have an important influence on the others. This calculation is then an inverse PageRank process, in which the steady-state of the Markov chain is known and given in Eq. (7), and the transition probabilities are searched.

3.2.1. *Algorithm to compute the connectivity matrix*

As it has been done in [25], the algorithm to compute the Markov transition matrix, that is in our case the stochastic connectivity matrix defining the influence of all the particles on the others, is given by the following steps

Step 1: The choice of an initial matrix $\boldsymbol{C}_0$. In our case, the initial matrix is random, but each line is then normalized, because the sum of all the terms in each *line* has to be 1.

Step 2: An iterative process is performed to adjust the entries of $\boldsymbol{C}_0$ in order to find a final transition matrix $\boldsymbol{C}_f$. The steady-state vector of $\boldsymbol{C}_f$ is the previously defined target vector $\boldsymbol{\pi}_{target}^T$. Let us define $\boldsymbol{C}_m$ the stochastic connectivity matrix during the step m of the iteration process, with the corresponding steady-state $\boldsymbol{\pi}_m^T$. Then the Markov process at time m can be described as

$$\boldsymbol{\pi}_m^T(\boldsymbol{C}_m - \boldsymbol{I}) = 0 \tag{9}$$

Algorithm 1 Algorithm to compute the stochastic connectivity matrix $\boldsymbol{C}$.

```
Starting with the initial connectivity matrix $\boldsymbol{C}_0$ ($m = 0$), calculate the residual $\boldsymbol{r}_m$ at step m. The first steady-state vector $\boldsymbol{\pi}_0$ is given as $1/n$ for all its components.
Calculation of $\delta$ using Algorithm 2.
while $||\boldsymbol{r}_{m+1}||^2 > \epsilon_{PR}$ do
    Pick the column of $\boldsymbol{C}_m$ corresponding to the maximum entry of the residual $\boldsymbol{r}_m$.
    Pick the column of $\boldsymbol{C}_m$ corresponding to the minimum entry of the residual $\boldsymbol{r}_m$.
    Pick a random row of $\boldsymbol{C}_m$.
    Check if the application of $\delta$ on the chosen row could alternate the positivity of all terms in the matrix $\boldsymbol{C}_m$. Check also if the application of $\delta$ does not keep the elements of $\boldsymbol{C}$ in $[0; 1]$. If it is the case, pick another row.
    Increase the entry of $\boldsymbol{C}_m$ selected in step (2) by $\delta$. Decrease the entry of $\boldsymbol{C}_m$ by $\delta$. This is the new connectivity matrix $\boldsymbol{C}_{m+1}$.
    Calculate the new steady state vector $\boldsymbol{\pi}^T_{m+1}$ corresponding to $\boldsymbol{C}_{m+1}$ using Eq. (A.4).
    Calculate the new residual $\boldsymbol{r}_{m+1}$ using Eq. (11).
end while
```

As said previously, the purpose of this calculation is to find the entries of $\boldsymbol{C}_m$ so that we have

$$\boldsymbol{\pi}^T_{target}(\boldsymbol{C}_m - \boldsymbol{I}) = 0 \tag{10}$$

that is $||\boldsymbol{\pi}^T_m - \boldsymbol{\pi}^T_{target}||^2 = 0$. Then, a residual $\boldsymbol{r}_m$ can be defined at the iteration m, which is

$$\boldsymbol{r}_m \equiv (\boldsymbol{\pi}^T_{target} - \boldsymbol{\pi}_m)(\boldsymbol{C}_m - \boldsymbol{I}) \tag{11}$$

Finally, the goal of this work is to find the components of $\boldsymbol{C}_m$ so that $||\boldsymbol{r}_m||^2 \leq \epsilon_{PR} \ll 1$, where ϵ_{PR} is the convergence threshold of the calculation. To do this, the components of $\boldsymbol{C}_m$ are adjusted at each iteration m by the factor δ, according to Algorithm 1 [25].

In the final converged connectivity matrix $\boldsymbol{C}_f$, the nonzero elements of row k are relative to the links going out of the page k, whereas the nonzero elements of column k are relative to the links coming in the page k. Then, in our case, the nonzero elements of column k show how the particle k influences the others, whereas the nonzero elements of row k show how the particle k is influenced by the others. As the sum of all the terms in each line of $\boldsymbol{C}$ is , one can note that the total influence of all the particles on one of them is always 1.

It is important to note that the changing parameters δ is defined as a function of the target vector $\boldsymbol{\pi}^T_{target}$. In fact, δ is the order of magnitude of the minimum component of $\boldsymbol{\pi}^T_{target}$. Then, δ is calculated according to Algorithm 2.

Because there are lots of random parameters in the algorithm, the final matrix C_f can be slightly different, from one calculation to another, even though the initial matrix $\boldsymbol{C}_0$ is the same [25]. Indeed, the final matrix depends on the randomly chosen row to be modified. Newton et al. [25] performed a statistical study to show the differences of the final matrices $\boldsymbol{C}_m$, which are all conditioned by the same initial matrix $\boldsymbol{C}_0$. It has been shown that the sensitivity of the final

Algorithm 2 Algorithm to compute the factor δ.

```
$ii = min(\boldsymbol{\pi}^T_{target})$
order magnitude = 0
while $ii \leq 1$ do
    $ii = ii \times 10$
    order magnitude = order magnitude + 1
end while
$\delta = 1 \times 10^{-\text{order magnitude}}$
```

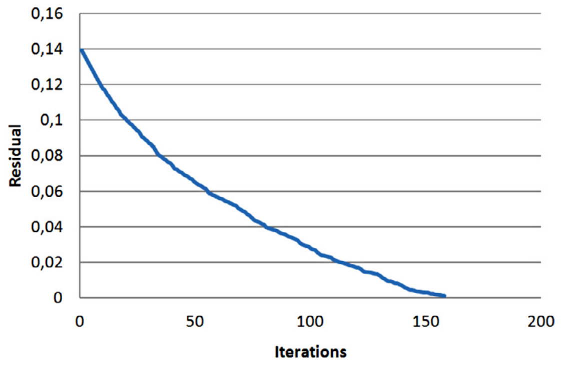

Fig. 6. Convergence of the residual r_m during the iterations.

converged connectivity matrix $\boldsymbol{C}_f$ with respect to the initial connectivity matrix $\boldsymbol{C}_0$ could be neglected (the order of magnitude of the standard deviation is 10 at the outside).

Finally, this calculation allows us to find a stochastic connectivity between all the particles of the swarm. The weighted influence between the particles, corresponding to the normalized target PageRank vector $\boldsymbol{\pi}^T_{target}$ is defined in Eq. (8).

3.2.2. Examples of calculations performed and issues

Some examples have been performed to show how the connectivity matrix $\boldsymbol{C}$ is calculated by Algorithm 1. The first example is the following : the target vector $\boldsymbol{\pi}^T_{target}$ is given by $\boldsymbol{\pi}^T_{target} = \begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix}$ and then the normalized vector is $\boldsymbol{\pi}^T_{target} = \begin{bmatrix}0.0667 & 0.1333 & 0.2000 & 0.6000\end{bmatrix}$. The first initial matrix $\boldsymbol{C}_0$ is random, and each line is normalized so that the sum of all the terms in each line is 1. Using Algorithm 1, the final PageRank vector is $\boldsymbol{\pi}^T_m = \begin{bmatrix}0.0737 & 0.1435 & 0.2099 & 0.5729\end{bmatrix}$ and we have $\left\|\boldsymbol{\pi}^T_{target}\right\| - \left\|\boldsymbol{\pi}^T_m\right\| = 0.0314$. One can see here that the two vectors are quiet similar. The final population connectivity is given by Eq. (12), and the convergence of the residual r_m during the iterations is given in Fig. 6.

$$\begin{bmatrix} 0.0024 & 0.0981 & 0.1648 & 0.7347 \\ 0.0092 & 0.0098 & 0.2633 & 0.7178 \\ 0.0031 & 0.1443 & 0.2773 & 0.8470 \\ 0.1250 & 0.1826 & 0.2773 & 0.4151 \end{bmatrix} \tag{12}$$

One can note that the particle n^o4 is the most influent in the swarm, which is coherent because its target value in $\boldsymbol{\pi}^T_{target}$ is the upmost.

It is important to note that if the values in the target vector $\boldsymbol{\pi}^T_{target}$ are too far from each other (about some powers of ten), the calculation does not converge. Actually, in that case, the factor δ is too small to change efficiently the connectivity matrix. For example, if the target vector is given by $\boldsymbol{\pi}^T_{target} = \begin{bmatrix} 1 & 1.E^{-10} & 1.E^{-10} & 1.E^{-10} \end{bmatrix}$ before normalization, the final connectivity matrix is the same as the initial one, because δ is $1E^{-10}$ and can not change the connectivity matrix components sufficiently to converge to $\boldsymbol{\pi}^T_{target}$.

In the same way, if the components of $\boldsymbol{\pi}^T_{target}$ are the same, the final topology should be a GBEST topology, in which all the particles influence the others with the same weight. The connectivity matrix should be full of non-zero components which would be slightly the same. Nevertheless, in that case, the calculation does not converge at all, because $\boldsymbol{\pi}^T_{target} = \boldsymbol{\pi}^T_0$, and the first residual is then 0. Though, Algorithm 1 does not activate the loop because $\|r_{m+1}\|$ is directly inferior to ϵ_{PR}.

Algorithm 3 Global Inverse-PageRank-PSO algorithm.

```
Random definition of the particles' velocity
Random definition of the particles' position
Random definition of the normalized connectivity matrix
for $iteration_{PSO} = 1$ to $it_{PSO,MAX}$ do
    Calculation of the fitness
    Calculation of the target vector using eq. (7).
    Random definition of the first connectivity matrix $\mathbf{C}_0$
    Calculation of the first residual $\mathbf{r}_m$
    while $(||\mathbf{r}_m|| > \epsilon)$ AND $(iteration_{PR} \le it_{PR,MAX})$ do
        Research of the best connectivity matrix $\mathbf{C}$ using Algorithm 1
    end while
    Updating of all $P_{i,best}^{iteration_{PSO}}$
    Updating of the best performance found so far by all the swarm $\mathbf{G}_{Best}$ and its fitness
    Calculation of the new speed of the particles using eq. (13)
    Calculation of the new position of the particles using eq. (13)
end for
```

3.3. Definition of the newly developed Inverse-PageRank-PSO

In I-PR-PSO, all the particles are used to influence each others, but their respective influences are weighted by the components of the previously seen stochastic connectivity matrix $\mathbf{C}$. Then, the position change of each swarm's particle is then given in the following way

$$\begin{cases} \mathbf{V}_i^{t+1} = \omega \times \mathbf{V}_i^t + c_1 \times rand_1 \times (\mathbf{P}_{i,best}^{t+1} - \mathbf{X}_i^t) \\ + c_2 \times rand_2 \times \sum_{j=1}^{n} \mathbf{C}_{ij} \times \left[\mathbf{P}_{j,best}^{t+1} - \mathbf{X}_i^t\right] \\ \mathbf{X}_i^{t+1} = \mathbf{X}_i^t + \mathbf{V}_i^{t+1} \end{cases} \tag{13}$$

As we have previously seen, the particle i is influenced by all the particles of the swarm, and their respective influence are given by the components of the ith line of $\mathbf{C}$, that is $\mathbf{C}_{ij}\ \forall j$.

The global I-PR-PSO algorithm is described in Algorithm 3 in which $it_{PSO,MAX}$ is the maximum number of PSO iterations, and $it_{PR,MAX}$ is the maximum number of PageRank iterations, that is the iterations needed to calculate the stochastic connectivity matrix $\mathbf{C}$.

Concerning the issues previously presented in part (3.2.2), one can note that when the particles have the same fitness values (that is when the components of $\boldsymbol{\pi}_{target}^T$ are slightly the same), or when the particles have fitness values far from each other in the solution domain (that is when the components of $\boldsymbol{\pi}_{target}^T$ are very different (about some powers of ten)), the population topology is then given by the first random connectivity matrix $\mathbf{C}_0$. This strategy corresponds to the one proposed by Mohais et al. in [12], in which it has been suggested that random topologies can be competitive to predefined ones [30]. Moreover, it has been shown in the literature that the proximity of individuals could cause premature convergence problems, because of the loss of diversity. This random re-actualization of the population topology is a solution to this loss of diversity.

Table 1
Calculation parameters.

Number of particles	50
Inertia weight ω	0.8
Acceleration constant c_1	2
Acceleration constant c_2	2
Maximum number of PSO iterations	600
Convergence threshold of the PageRank algorithm ϵ_{PR}	$1E-03$
Maximum number of PageRank iterations	6000
Dimension of the problem	10, 20, 30 and 50

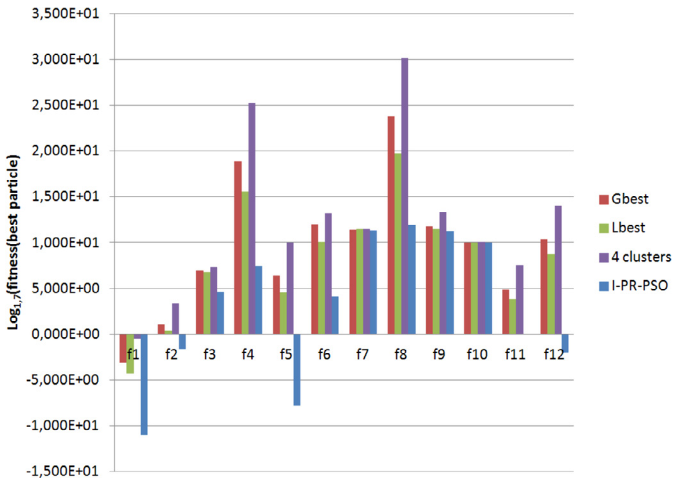

Fig. 7. Dimension 10.

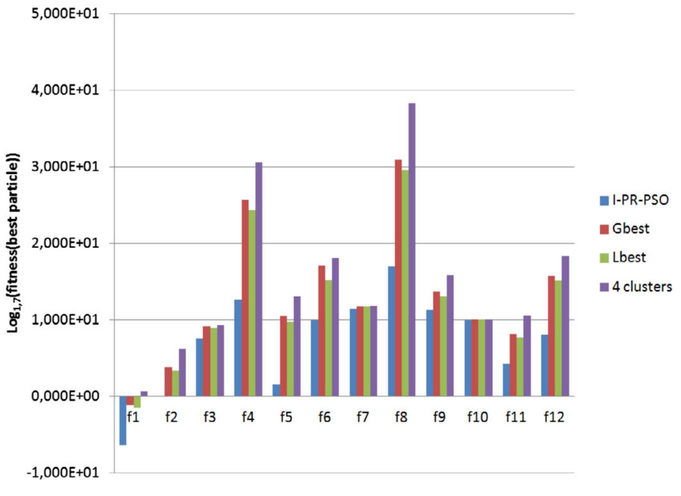

Fig. 8. Dimension 20.

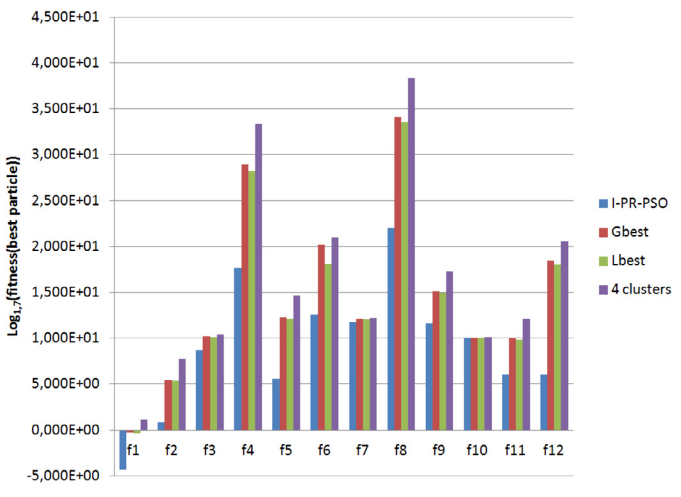

Fig. 9. Dimension 30.

4. Simulation results

4.1. PSO parameters and benchmark functions

I-PR-PSO has been tested on the different benchmark functions given in Table 2 in which D represents the dimension of the problem.

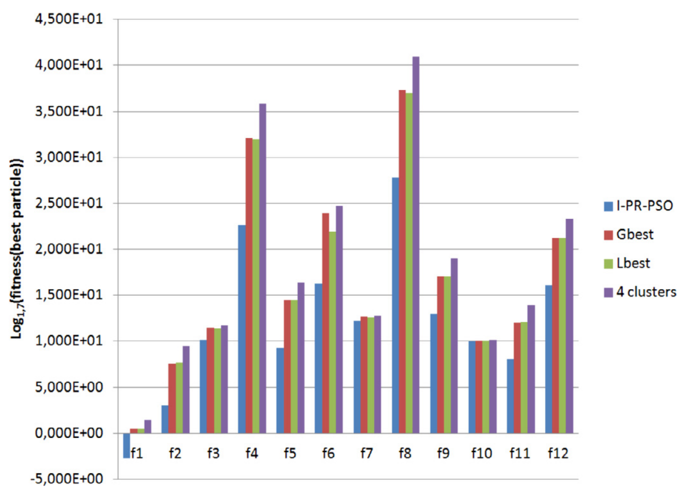

Fig. 10. Dimension 50 (Reached values of the tested objective functions after 600 PSO iterations (mean of the 100 runs, in a log scale)).

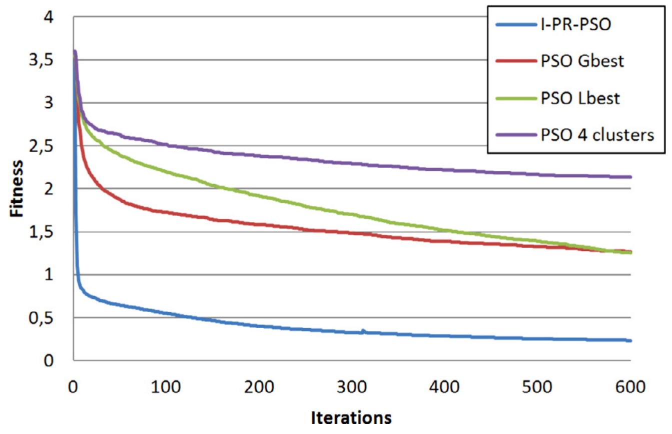

Fig. 11. Ackley.

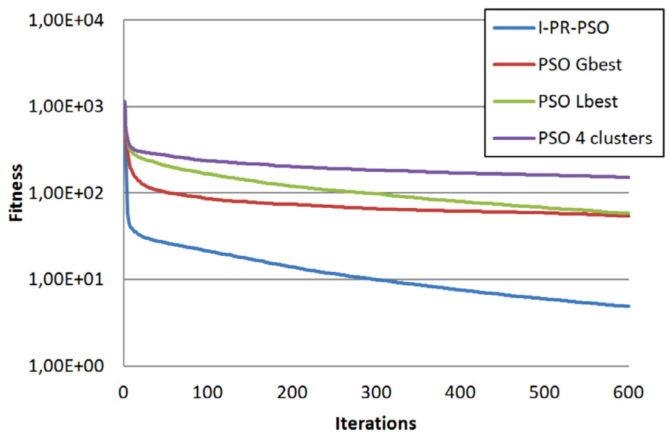

Fig. 12. Griewank.

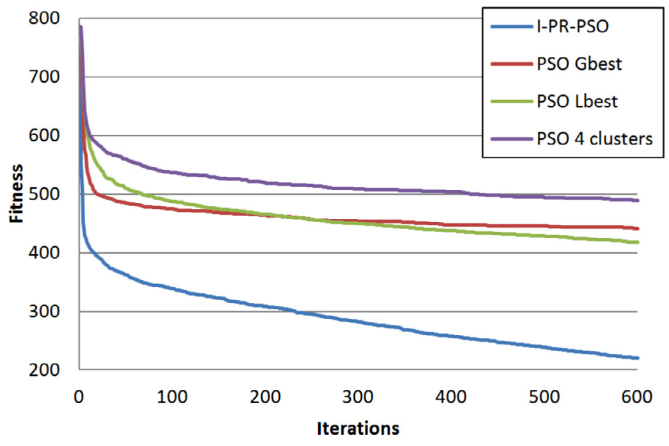

Fig. 13. Rastrigin.

A statistical study has been performed to obtain sufficient results to prove the efficiency of I-PR-PSO. On each function, 100 runs have been performed in the dimensions 10, 20, 30 and 50, with 50 particles. As it has been shown in the literature that PSO can be more efficient than other metaheuristic methods in large dimensions [31,32], this research focuses especially on the comparison between different PSO variants. Then, I-PR-PSO has been compared to three different versions of classic PSO, that is with the previously presented GBEST topology, the LBEST topology, and the 4-clusters topology, with the same calculation parameters given in Table 1.

4.2. Obtained results

In this paper, the best value of the objective function reached after 600 PSO iterations is investigated, for all the twelve objective

Table 2
Benchmark functions.

	Function	Mathematical expression	Opt. pos.	Opt. val.	Type	Domain	V_{max}
f1	Ackley	$20+e-20e^{-0.2\sqrt{\frac{\sum(x_{i-1}^D)^2}{D}}}-e^{\frac{\sum_{i=1}^D cos(2\pi x_i)}{D}}$	(0, 0,..., 0)	0	Multimodal	[−1; 1]	1
f2	Griewank	$1+\frac{\sum_{i=1}^D (x_i-100)^2}{4000}-\prod_{i=1}^D cos(\frac{x_i-100}{\sqrt{i}})$	(0, 0,..., 0)	0	Multimodal	[−600; 600]	500
f3	Rastrigin	$\sum_{i=1}^D x_i^2-10cos(2\pi x_i)+10D$	(0, 0,..., 0)	0	Multimodal	[−5.12; 5.12]	5
f4	Rosenbrock	$\sum_{i=1}^{D-1} 100(x_{i+1}-x_i)^2+(x_i+1)^2$	(1, 1,..., 1)	0	Unimodal	[−50; 50]	50
f5	Sphere	$\sum_i^D x_i^2$	(0, 0,..., 0)	0	Unimodal	[−50; 50]	50
f6	Rotate hyper ellips.	$\sum_{i=1}^D (\sum_{j=1}^i x_j)^2$	(0, 0,..., 0)	0	Multimodal	[−65.536; 65.536]	65
f7	Shifted Rastrigin	$\sum_{i=1}^D ((x_i-1)^2-10cos(2\pi(x_i-1))+10)+390$	(1, 1,..., 1)	390	Multimodal	[−5; 5]	5
f8	Shifted Rosenbrock	$\sum_{i=1}^{D-1} (100((x_{i+1}-1)-(x_i-1)^2)^2+(x_{i-1}-1)^2)+390$	(1, 1,..., 1)	390	Multimodal	[−100; 100]	100
f9	Shifted Sphere	$\sum_{i=1}^D ((x_i-1)^D)+400$	(1, 1,...1,)	450	Multimodal	[−100; 100]	100
f10	Shifted Ackley	$-20exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^D (x_i-1)^2})-exp(\frac{1}{D}\sum_{i=1}^D cos(12\pi(x_i-1)))$	(1, 1,..., 1)	200	Multimodal	[−32; 32]	32
f11	Bohachevsky	$\sum_{i=1}^{D-1} (x_i^{12}+2x_{i+1}^2-0.3cos(3\pi x_i)-0.4cos(4\pi x_{i+1})+0.7$	(0, 0,..., 0)	0	Unimodal	[−15; 15]	15
f12	Schwefel's problem 1.2	$\sum_{i=1}^D (\sum_{j=1}^i x_j)^2$	(0, 0,..., 0)	0	Unimodal	[−65.536; 65.536]	65.536

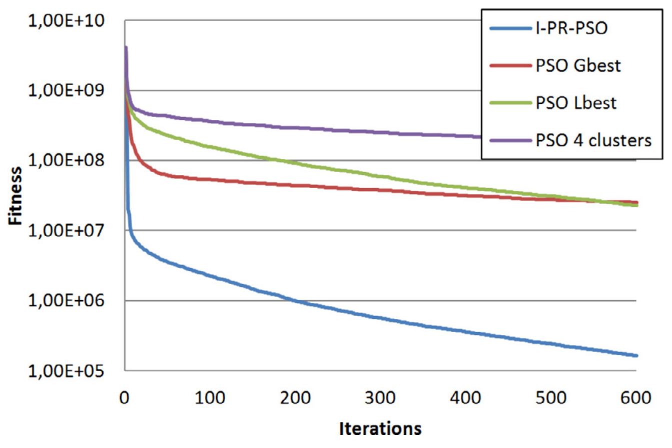

Fig. 14. Rosenbrock.

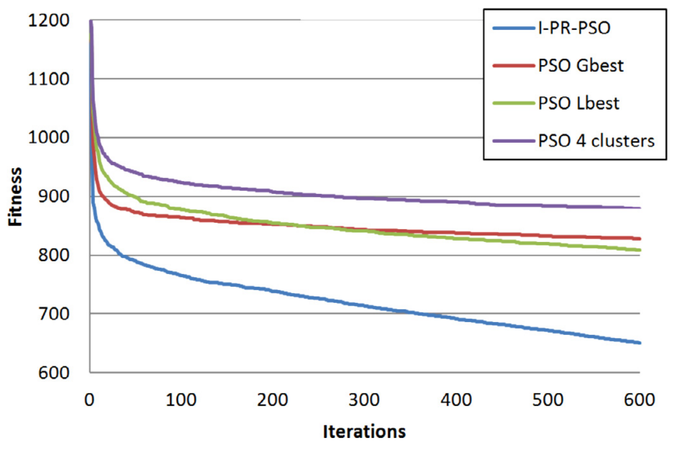

Fig. 17. Shifted Rastrigin.

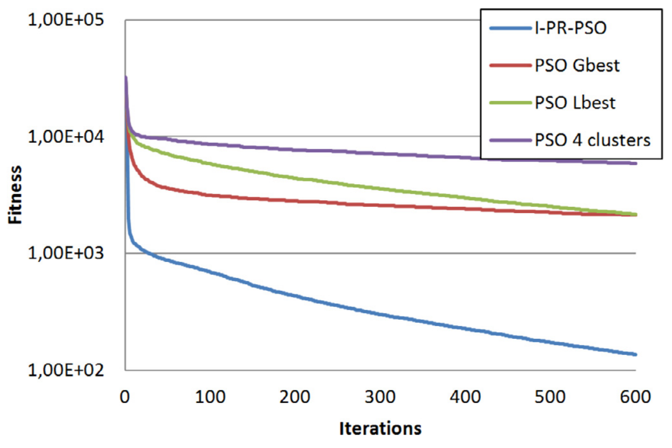

Fig. 15. Sphere.

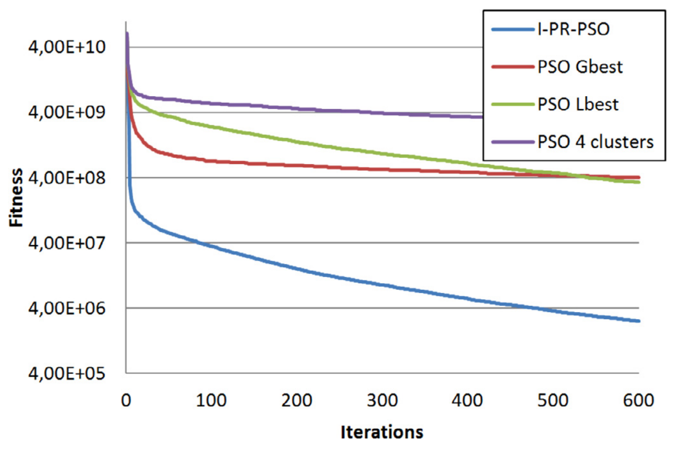

Fig. 18. Shifted Rosenbrock.

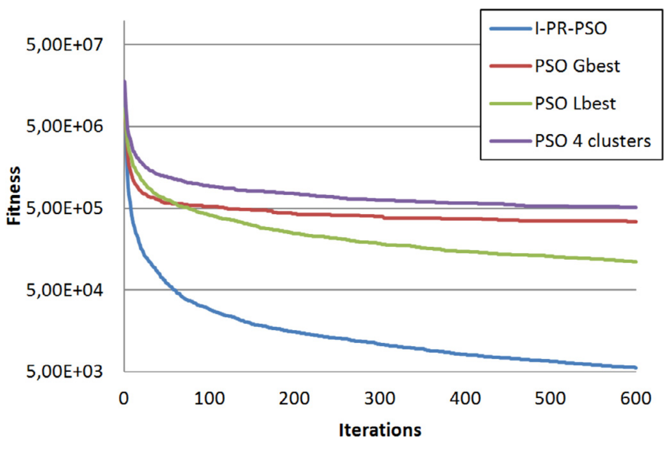

Fig. 16. Rotate hyper ellipsoid function (convergence curves of 50 dimensional problems (mean of the 100 runs)).

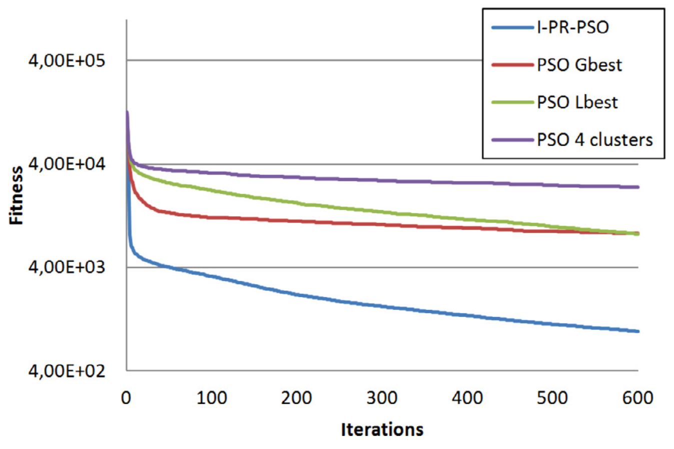

Fig. 19. Shifted Sphere.

functions, with all the 4 different PSO variants. The results, that is the best values of the objective function found so far, are presented in a log scale in Figs. 7–10. The values obtained are given in Tables 3 and 4 in which the mean and the standard deviation of all the 100 runs are presented.

Based on the results given in Figs. 7–10, we conclude that our proposed I-PR-PSO is more efficient than the tested peers on the tested objective functions in dimensions 10, 20, 30 and 50.

Finally, to have a visual aspect of the convergence of the different algorithms on the considered objective functions, the mean of the best fitness values found over the iterations, for the 100 different calculations performed, is presented. The convergence curves are presented for the dimension 50 in Figs. 11–22. Some of these graphs are presented in a log scale, so that the results are readable.

Table 3
100 runs: reached values after 600 PSO iterations (mean ± st. dev.).

	Inverse-PageRank-PSO	PSO GBest	PSO Lbest	PSO 4-clusters
Dimension 10				
f1	**2.98E-03** ± 1.263E-03	1.95E-01 ± 5.79E-02	1.01E-01 ± 3.26E-02	7.68E-01 ± 1.24E-01
f2	**4.26E-01** ± 1.19E-01	1.77E+00 ± 3.48E-01	1.21E+00 ± 1.68E-01	6.03E+00 ± 1.46E+00
f3	**1.17E+01** ± 6.86E+00	4.06E+01 ± 6.04E+00	3.65E+01 ± 6.04E+00	4.91E+01 ± 6.51E+00
f4	**5.23E+01** ± 9.39E+01	2.23E+04 ± 1.62E+04	3.96E+03 ± 3.31E+03	6.74E+05 ± 3.62E+05
f5	**1.59E-02** ± 1.57E-02	2.97E+01 ± 1.36E+01	1.14E+01 ± 6.34E+00	2.01E+02 ± 5.64E+01
f6	**8.89E+00** ± 7.55E+00	5.93E+02 ± 1.98E+02	2.09E+02 ± 8.56E+01	1.11E+03 ± 3.78E+02
f7	**4.06E+02** ± 4.93E+00	4.30E+02 ± 5.32E+00	4.37E+02 ± 7.39E+00	4.37E+02 ± 7.39E+00
f8	**5.68E+02** ± 4.13E+02	2.98E+05 ± 2.79E+05	3.58E+04 ± 3.88E+04	9.03E+06 ± 5.02E+06
f9	**4.00E+02** ± 7.80E-02	5.12E+02 ± 5.82E+01	4.42E+02 ± 2.04E+01	1.19E+03 ± 2.14E+02
f10	**2.00E+02** ± 3.80E-01	2.06E+02 ± 8.00E-01	2.04E+02 ± 6.43E-01	2.10E+02 ± 8.87E-01
f11	**9.95E-01** ± 6.39E-01	1.32E+01 ± 3.32E+00	7.47E+00 ± 1.90E+00	5.46E+01 ± 1.30E+01
f12	**3.48E-01** ± 3.87E-01	2.43E+02 ± 1.06E+02	1.02E+02 ± 6.19E+01	1.73E+03 ± 5.34E+02
Dimension 20				
f1	**3.39E-02** ± 1.22E-02	5.51E-01 ± 1.08E-01	4.58E-01 ± 9.59E-02	1.43E+00 ± 1.72E-01
f2	**1.06E+00** ± 5.36E-02	7.44E+00 ± 1.74E+00	5.91E+00 ± 1.46E+00	2.69E+01 ± 5.22E+00
f3	**5.62E+01** ± 2.55E+01	1.29E+02 ± 1.02E+01	1.17E+02 ± 1.37E+01	1.46E+02 ± 1.09E+01
f4	**8.24E+02** ± 8.21E+02	8.22E+05 ± 4.50E+05	4.17E+05 ± 2.51E+05	1.14E+07 ± 4.21E+06
f5	**2.29E+00** ± 1.34E+00	2.62E+02 ± 7.46E+01	1.85E+02 ± 4.96E+01	1.03E+03 ± 2.13E+02
f6	**1.97E+02** ± 9.46E+01	9.00E+03 ± 2.19E+03	3.32E+03 ± 1.29E+03	1.52E+04 ± 3.86E+03
f7	**4.52E+02** ± 2.57E+01	5.19E+02 ± 1.03E+01	5.11E+02 ± 1.35E+01	5.38E+02 ± 1.10E+01
f8	**8.35E+03** ± 1.28E+04	1.37E+07 ± 7.24E+06	6.81E+06 ± 3.91E+06	2.92E+08 ± 2.71E+08
f9	**4.09E+02** ± 5.00E+00	1.47E+03 ± 2.96E+02	1.07E+03 ± 2.10E+02	4.47E+03 ± 7.34E+02
f10	**2.03E+02** ± 4.89E-01	2.09E-02 ± 8.66E-01	2.08E+02 ± 8.03E-01	2.14E+02 ± 6.33E-01
f11	**9.76E+00** ± 2.35E+00	7.87E+01 ± 1.72E+01	6.15E+01 ± 1.51E+01	2.75E+02 ± 4.93E+01
f12	**7.25E+01** ± 4.12E+01	4.24E+03 ± 1.11E+03	3.15E+03 ± 9.79E+02	1.74E+04 ± 2.88E+03

Table 4
100 runs: reached values after 600 PSO iterations (mean ± st. dev.).

	Inverse-PageRank-PSO	PSO GBest	PSO Lbest	PSO 4-clusters
Dimension 30				
f1	**1.01E-01** ± 2.69E-02	8.57E-01 ± 1.46E-01	8.18E-01 ± 1.27E-01	1.78E+00 ± 1.63E-01
f2	**1.52E+00** ± 2.33E-01	1.78E+01 ± 3.52E+00	1.74E+01 ± 3.93E+00	6.00E+01 ± 9.33E+00
f3	**1.02E+02** ± 4.39E+01	2.27E+02 ± 1.40E+01	2.17E+02 ± 1.93E+01	2.55E+02 ± 1.46E+01
f4	**1.14E+04** ± 9.16E+03	4.63E+06 ± 2.08E+06	3.20E+06 ± 1.49E+06	4.77E+07 ± 1.47E+07
f5	**1.97E+01** ± 7.62E+00	6.86E+02 ± 1.54E+02	6.14E+02 ± 1.46E+02	2.36E+03 ± 4.15E+02
f6	**7.98E+02** ± 4.15E+02	4.41E+04 ± 1.31E+04	1.49E+04 ± 5.18E+03	6.95E+04 ± 2.12E+04
f7	**5.12E+02** ± 4.02E+01	6.18E+02 ± 1.40E+01	6.40E+02 ± 1.98E+01	6.46E+02 ± 1.51E+01
f8	**1.19E+05** ± 9.57E+04	7.07E+07 ± 3.37E+07	5.34E+07 ± 2.60E+07	6.92E+08 ± 2.71E+08
f9	**4.76E+02** ± 2.91E+01	3.05E+03 ± 6.11E+02	2.76E+03 ± 5.17E+02	9.68E+03 ± 1.38E+03
f10	**2.04E+02** ± 5.41E-01	2.11E+02 ± 9.36E-01	2.11E+02 ± 8.24E-01	2.15E+02 ± 5.88E-01
f11	**2.43E+01** ± 5.45E+00	2.07E+02 ± 4.13E+01	1.83E+02 ± 3.67E+01	6.21E+02 ± 1.00E+02
f12	**2.43E+01** ± 4.45E+00	1.77E+04 ± 4.05E+03	1.42E+04 ± 3.24E+03	5.48E+04 ± 8.89E+03
Dimension 50				
f1	**2.35E-01** ± 5.10E-02	1.27E+00 ± 1.90E-01	1.26E+00 ± 1.65E-01	2.13E+00 ± 1.25E-01
f2	**4.90E+00** ± 1.18E+00	5.41E+01 ± 1.03E+01	5.79E+01 ± 1.13E+01	1.52E+02 ± 1.89E+01
f3	**2.20E+02** ± 7.53E+01	4.41E+02 ± 1.82E+01	4.18E+02 ± 2.73E+01	4.90E+02 ± 2.21E+01
f4	**1.67E+05** ± 8.13E+04	2.53E+07 ± 9.20E+06	2.29E+07 ± 8.61E+06	1.79E+08 ± 4.93E+07
f5	**1.35E+02** ± 3.80E+01	2.13E+03 ± 3.83E+02	2.15E+03 ± 3.59E+02	5.91E+03 ± 8.87E+02
f6	**5.59E+03** ± 3.09E+03	3.21E+05 ± 8.21E+04	1.10E+05 ± 4.17E+04	5.10E+05 ± 1.59E+05
f7	**6.51E+02** ± 7.02E+01	8.28E+02 ± 1.93E+01	8.08E+02 ± 2.60E+01	8.79E+02 ± 2.29E+01
f8	**2.52E+06** ± 1.47E+06	4.01E+08 ± 1.32E+08	3.41E+08 ± 1.23E+08	2.77E+089 ± 7.59E+08
f9	**9.66E+02** ± 1.66E+02	8.52E+03 ± 1.45E+03	8.44E+03 ± 1.40E+03	2.38E+04 ± 3.03E+03
f10	**2.00E+02** ± 3.80E-01	2.13E+02 ± 5.95E-01	2.13E+02 ± 7.30E-01	2.17E+02 ± 4.22E-01
f11	**7.28E+01** ± 1.63E+01	5.87E+02 ± 1.11E+02	6.14E+02 ± 9.87E+01	1.63E+03 ± 2.46E+02
f12	**5.18E+03** ± 1.53E+03	7.96E+04 ± 1.49E+04	7.94E+04 ± 1.42E+04	2.34E+05 ± 3.12E+04

As one can see in these figures, the results presented are coherent with those previously presented. The Inverse-PageRank-PSO algorithm converges closer to the global optimum than its tested peers, in all dimensions. Moreover, I-PR-PSO is quicker to converge than the tested peers. Actually, the algorithm has a better global research ability, while its local research ability is not better than the other algorithms: once the swarm is close to the global optimum of the tested objective function, Inverse-PageRank-PSO needs lots of iterations to finally stabilize the swarm. I-PR-PSO is also quicker to converge and converges closer to the global optimum than its tested peers in dimensions 10, 20 and 30, but the convergence curves are not presented here.

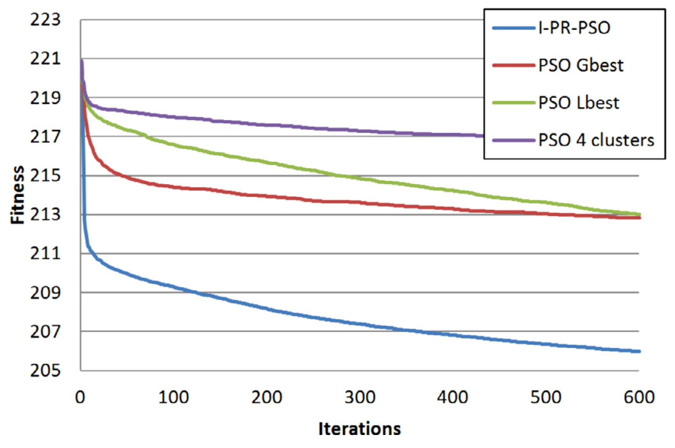

Fig. 20. Shifted Ackley.

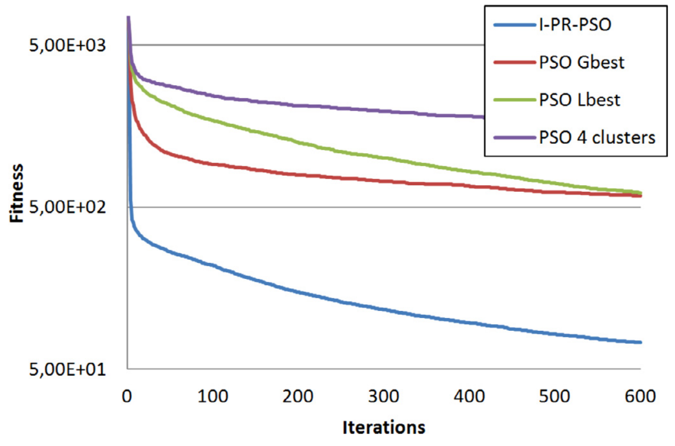

Fig. 21. Bohachevsky.

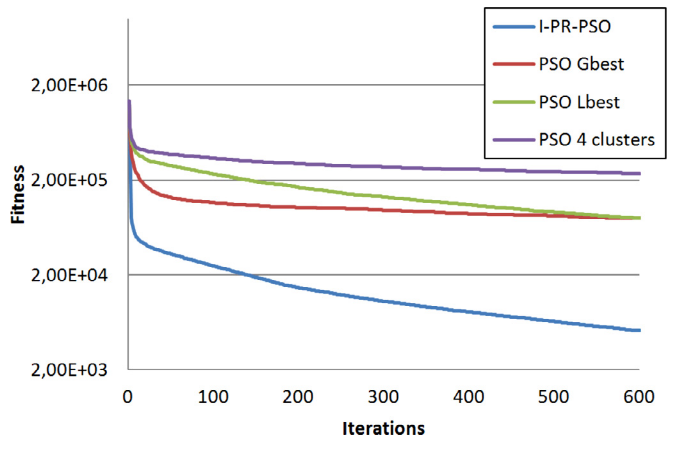

Fig. 22. Schwefel's problem 1.2 (convergence curves of 50 dimensional problems (mean of the 100 runs)).

5. Discussion and conclusions

In this paper, I-PR-PSO is proposed to solve unconstrained minimization optimization problems defined in continuous solution domains. In I-PR-PSO, the population topology evolves during the iterations. The population developed is a weighted GBEST topology, in which the weights are defined using the Markov chains theory, and the Inverse PageRank algorithm in particular. Indeed, Inverse-PageRank-PSO provides a general adaptive algorithm that updates the population topology of the swarm, without any additional parameter compared to classical PSO. Its social behavior is then enhanced based on the actual evolution of the population.

The obtained numerical results show that I-PR-PSO has the ability to find the global optimum of the considered objective function than its peers. So I-PR-PSO achieves a better balance between the exploration and exploitation phases needed by the particles to find the global optimum in large dimensions. Nevertheless, the algorithm has a better global research ability than its peers, while its local research ability is not better than the other algorithms: once the swarm is close to the global optimum of the tested objective function, I-PR-PSO needs lots of iterations to finally stabilize the swarm.

Moreover, I-PR-PSO is quicker to converge than the tested peers in terms of number of objective function evaluations needed to converge. Comparing I-PR-PSO with the work of Lim and Isa [33], we can show that I-PR-PSO has a better ability to push the swarm close to the global optimum (I-PR-PSO needs approximately 1000 function evaluations while its peers need approximately 5000 function evaluations on the same objective functions). However, an additional iterative process is needed to calculate the connectivity matrix $\mathbf{C}$. I-PR-PSO is then much more longer than its peers to converge in terms of CPU time. Thus, I-PR-PSO is very efficient in mechanical applications when the Finite Element Method is used because, in this context, the evaluations of the cost function are very expensive. Then, reducing the number of calls to the objective function could also reduce efficiently the CPU time. On the contrary, if the objective function evaluations are not very expensive in terms of CPU time, I-PR-PSO could be more expensive than its peers, but could find better results, as it has been seen in Figs. 7–10 and 17–22.

Obviously, the No Free Lunch theorem has shown that no algorithm can perform better than any other, on all possible objective function [19,34]. Then, testing I-PR-PSO on different benchmark functions that have been identified as hard problems can show that this newly developed algorithm could be more efficient on lots of different objective functions. This algorithm has been tested on engineering structural optimization problems [35], and has been shown to be very efficient on constrained optimization problems.

Acknowledgment

Computations have been performed on the supercomputer facilities of the Mésocentre de calcul de FrancheComté.

Appendix. The PageRank algorithm

As given in the literature, the PageRank of a page P_k, noted $PR(P_k)$, is the sum of the PageRanks of all pages P_l pointing into P_k normed by the number of outgoing links from P_l, as one can see in Eq. (A.1).

$$PR(P_k) = \sum_{P_l \in B_{P_k}} \frac{PR(P_l)}{|P_l|} \tag{A.1}$$

where B_{P_k} represents the set of all the pages pointing to the page P_k, and $|P_l|$ is the number of links outing P_l. Then, it can be easily understood that an iterative process is required to solve this problem, since the PageRanks of the pages P_l are not known at the beginning of the calculation. Eq. (A.1) is successively applied and the PageRanks are updated at each iteration to effectively compute the PageRanks of all the considered pages. Noting $PR_{m+1}(P_k)$ the PageRank of P_k at iteration $m+1$, the iterative process is given as

$$PR_{m+1}(P_k) \Leftarrow \sum_{P_l \in B_{P_k}} \frac{PR_m(P_l)}{|P_l|} \tag{A.2}$$

This process starts with $PR_0(P_k) = 1/n$ for all pages P_k where n is the number of webpages in the collection, and the previous iterative process is achieved until some stable values are found. Using a matrix notation, Brin and Page have greatly simplified and improved these calculations. At each iteration, a PageRank vector, noted $\boldsymbol{\pi}^T$ is computed, which is a $1 \times n$ vector holding all the PageRank values for all the pages of the web. A $n \times n$ matrix $\boldsymbol{C}$, which is the stochastic probabilistic adjacency matrix of the graph, is defined. This matrix is a row normalized hyperlink matrix [36] given by

$$\begin{cases} C_{kl} = \frac{1}{|P_k|} & \text{if there is a link from node } k \text{ to node } l \\ 0 & \text{otherwise} \end{cases} \tag{A.3}$$

One can notice that the nonzero elements of row k are relative to the links going out the page k, whereas the nonzero elements of column k are relative to the links coming in the page k. Moreover, as the components of $\boldsymbol{C}$ are normalized, the sum of all the terms in each line is 1, as it has been seen in Eq. (4).

With this notation, the Markov dynamical model given in Eq. (A.2) can be written as

$$\boldsymbol{\pi}^{(m+1)T} = \boldsymbol{\pi}^{(m)T}\boldsymbol{C} \tag{A.4}$$

where $\boldsymbol{\pi}^{(m+1)T}$ and $\boldsymbol{\pi}^{(m)T}$ are the state vectors of the Markov chain at discrete times m and $m + 1$, respectively.

Two numerical methods have been developed to solve the PageRank Problem. The first one is the solving of the following *eigenvector* problem for $\boldsymbol{\pi}^T$

$$\begin{cases} \boldsymbol{\pi}^T = \boldsymbol{\pi}^T\boldsymbol{C} \\ \boldsymbol{\pi}^T\boldsymbol{e} = 1 \end{cases} \tag{A.5}$$

The second one is the solving of the following *linear homogeneous system* for $\boldsymbol{\pi}^T$

$$\begin{cases} \boldsymbol{\pi}^T(\boldsymbol{I} - \boldsymbol{C}) = \boldsymbol{0}^T \\ \boldsymbol{\pi}^T\boldsymbol{e} = 1 \end{cases} \tag{A.6}$$

where $\boldsymbol{I}$ is the dimension-n identity matrix, and $\boldsymbol{e}^T$ is the row vector of all ones.

In the first case, the goal is to find the normalized dominant left-hand eigenvector of $\boldsymbol{C}$, corresponding to the dominant eigenvalue $\lambda_1 = 1$. In the second case, the goal is to find the normalized left-hand null vector of $\boldsymbol{I} - \boldsymbol{C}$. In both, the second equation $\boldsymbol{\pi}^T\boldsymbol{e} = 1$ insures that $\boldsymbol{\pi}^T$ is a probability vector. These observations allow us to calculate directly the steady state distribution $\boldsymbol{\pi}^T$ from the stochastic connectivity matrix $\boldsymbol{C}$. As it has been seen in Eq. (A.3), the sum of all elements of the rows of $\boldsymbol{C}$ is equal to 1, that is why there is always at least one eigenvalue equals to 1.

References

[1] Brin S, Page L. The anatomy of a large-scale hypertextual web search engine. In: Proceedings of the seventh international world wide web conference. Comput Netw ISDN Syst 1998;30(1):107–17.

[2] Dorigo M, Birattari M, Stutzle T. Ant colony optimization. Comput Intell Mag IEEE 2006;1(4):28–39.

[3] Goldberg D. Genetic algorithms in search, optimization and machine learning. Addison-Wesley; 1989.

[4] Fonseca CM, Fleming PJ. An overview of evolutionary algorithms in multiobjective optimization. Evolut Comput 1995;3:1–16.

[5] Coello C, Lamont G, Veldhuizen DV. Evolutionary algorithms for solving multi-objective problems (genetic and evolutionary computation). Secaucus, NJ, USA: Springer-Verlag New York, Inc.; 2006.

[6] Reynolds C. Flocks, herds and schools: a distributed behavioral model. In: Proceedings of the 14th annual conference on computer graphics and interactive techniques, SIGGRAPH '87. New York, NY, USA: ACM; 1987. p. 25–34.

[7] Kennedy J, Eberhart R. Particle swarm optimization. In: Proceedings of the IEEE international conference on neural networks, 4; 1995. p. 1942–8.

[8] Akbari R, Ziarati K. A rank based particle swarm optimization algorithm with dynamic adaptation. J Comput Appl Math 2011;235(8):2694–714.

[9] Eberhart R, Kennedy J. A new optimizer using particle swarm theory. In: Proceedings of the sixth international symposium on micro machine and human science, MHS '95.; 1995. p. 39–43.

[10] Jiang Y, Hu T, Huang C, Wu X. An improved particle swarm optimization algorithm. Appl Math Comput 2007;193(1):231–9.

[11] Angeline P. Using selection to improve particle swarm optimization. In: Proceedings of the 1998 IEEE international conference on evolutionary computation proceedings, IEEE world congress on computational intelligence; 1998. p. 84–9.

[12] Mohais A, Mendes R, Ward C, Posthoff C. Neighborhood re-structuring in particle swarm optimization. In: Proceedings of the 18th Australian joint conference on artificial intelligence, LNCS 3809. Springer; 2005. p. 776–85.

[13] Suganthan P. Particle swarm optimiser with neighbourhood operator. In: Proceedings of the 1999 congress on evolutionary computation, CEC 99, 3; 1999. p. 1962.

[14] Pasupuleti S, Battiti R. The gregarious particle swarm optimizer (G-PSO). In: Proceedings of the 8th annual conference on genetic and evolutionary computation, GECCO '06. ACM; 2006. p. 67–74.

[15] Janson S, Middendorf M. A hierarchical particle swarm optimizer. In: The 2003 congress on evolutionary computation, CEC '03, 2; 2003. p. 770–6.

[16] Løvbjerg M, Rasmussen T, Krink T. Hybrid particle swarm optimiser with breeding and subpopulations. In: Proceedings of the genetic and evolutionary computation conference, GECCO-2001. Morgan Kaufmann; 2001. p. 469–76.

[17] Vesterstrom J, Riget J, Krink T. Division of labor in particle swarm optimisation. In: Proceedings of the evolutionary computation, CEC 02. Washington, DC, USA: IEEE Computer Society; 2002. p. 1570–5.

[18] Blackwell T, Branke J. Multiswarms, exclusion, and anti-convergence in dynamic environments. IEEE Trans Evolut Comput 2006;10(4):459–72.

[19] Mendes R, Kennedy J, Neves J. Watch thy neighbor or how the swarm can learn from its environment. In: Proceedings of the 2003 IEEE swarm intelligence symposium SIS '03; 2003. p. 88–94.

[20] Mendes R, Kennedy J, Neves J. The fully informed particle swarm: simpler, maybe better. IEEE Trans Evolut Comput 2004;8:204–10.

[21] Holland J. Adaptation in natural and artificial systems: ASN introductory analysis with applications to biology, control, and artificial intelligence, 8. Oxford, England: U Michigan Press; 1975.

[22] Eberhart R, Shi Y. Tracking and optimizing dynamic systems with particle swarms. In: Proceedings of the 2001 congress on evolutionary computation, 1; 2001. p. 94–100.

[23] Luh G, Lin C, Lin Y. A binary particle swarm optimization for continuum structural topology optimization. Appl Soft Comput 2011;11(2):2833–44.

[24] Langville A, Meyer C. Deeper inside pagerank. Internet Math 2004;1.

[25] Newton P, Mason J, Bethel K, Bazhenova L, Nieva J, Kuhn P. A stochastic Markov chain model to describe lung cancer growth and metastasis. PLoS One 2012;7(4):e34637.

[26] Gzyl H, Velásquez Y. Reconstruction of transition probabilities by maximum entropy in the mean. In: Proceedings of the Bayesian inference and maximum entropy methods in science and engineering, 617. AIP Publishing; 2002. p. 192–203.

[27] Gzyl H, Velásquez Y. Maximum entropy in the mean: a useful tool for constrained linear problems. In: Proceedings of 22nd International Workshop on Bayesian inference and maximum entropy methods in science and engineering. American Institute of Physics; 2003. p. CP659.

[28] Csiszar I. Why least squares and maximum entropy? An axiomatic approach to inference for linear inverse problems. Ann Statist 1991;19(4):2032–66.

[29] Clerc M. Cooperation mechanisms in particle swarm optimisation. Tech. Rep. 2013:20. http://hal.archives-ouvertes.fr/hal-00868161.

[30] de Oca M, Stutzle T, Birattari M, Dorigo M. Frankenstein's PSO: a composite particle swarm optimization algorithm. IEEE Trans Evolut Comput 2009;13(5):1120–32.

[31] Kaveh A, Zolghadr A. Comparison of nine meta-heuristic algorithms for optimal design of truss structures with frequency constraints. Adv Eng Softw 2014;76(0):9–30.

[32] Lima CS, Lapa C, Pereira N, da Cunha J, Alvim A. Comparison of computational performance of GA and PSO optimization techniques when designing similar systems - typical PWR core case. Ann Nucl Energy 2011;38(6):1339–46.

[33] Lim WH, Isa NM. Particle swarm optimization with increasing topology connectivity. Eng Appl Artif Intell 2014;27:80–102.

[34] Wolpert D, Macready W. No free lunch theorems for optimization. IEEE Trans Evolut Comput 1997;1(1):67–82.

[35] DiCesare N, Lahrire A, Christ L, Chamoret D, Domaszewski M. Validation de la méthode d'optimisation par essaim particulaire basée sur un processus de Markov à temps discret (PageRank PSO) : Application à la mécanique. Proceedings of 12eme colloque national en calcul des structures, CSMA2015 2015.

[36] Langville A, Meyer C. Google's PageRank and beyond. The science of search engine rankings. Princeton University Press; 2012.