A Hierarchical Particle Swarm Optimizer and Its Adaptive Variant

Stefan Janson and Martin Middendorf, *Member, IEEE*

***Abstract*—A hierarchical version of the particle swarm optimization (PSO) metaheuristic is introduced in this paper. In the new method called H-PSO, the particles are arranged in a dynamic hierarchy that is used to define a neighborhood structure. Depending on the quality of their so-far best-found solution, the particles move up or down the hierarchy. This gives good particles that move up in the hierarchy a larger influence on the swarm. We introduce a variant of H-PSO, in which the shape of the hierarchy is dynamically adapted during the execution of the algorithm. Another variant is to assign different behavior to the individual particles with respect to their level in the hierarchy. H-PSO and its variants are tested on a commonly used set of optimization functions and are compared to PSO using different standard neighborhood schemes.**

***Index Terms*—Author, please supply your own keywords or send a blank e-mail to keywords@ieee.org to receive a list of suggested keywords.**

I. Introduction

THE particle swarm optimization (PSO) method for function optimization has been introduced by Kennedy and Eberhart in [1] and is inspired by the emergent motion of a flock of birds searching for food. Like in other optimization metaheuristics ([2]), as simulated annealing ([3], [4]), evolutionary algorithms ([5]–[8]), or ant colony optimization (ACO) ([9]–[11]), the search for an optimum is an iterative process that is based on random decisions. Another similarity of PSO to evolutionary algorithms and the population based version of ACO ([12]) is that a population of solutions or agents is used, which cooperate in finding better solutions. Evolutionary algorithms use principles of natural evolution like mutation, crossover and selection to obtain better solutions from the actual population of solutions. ACO is inspired by the foraging behavior of ants, which find short paths to food sources by marking their paths with pheromones. In ACO, a new solution is created by a simple agent, called ant, that uses a constructive solution generation method. The decisions of the ant are guided by artificial pheromone information that stems from former ants that have found good solutions. A PSO algorithm iteratively explores a multidimensional search space with a swarm of individuals, that are referred to as particles, looking for the global minimum (or maximum). Each particle "flies" through the search space according to its velocity vector. In every iteration, the velocity vector is adjusted so that prior personal successful positions (cognitive aspect) and the best position found by particles within a specific neighborhood (social aspect) act as attractors. In this paper we concentrate on PSO for continuous search spaces but it should be mentioned that PSO has also been applied to discrete optimization problems (e.g., [13]).

In the original PSO algorithm, the neighborhood of a particle consists of all particles, so that the global best position, i.e., the best solution found so far, directly influences its behavior. Several authors have investigated the use of restricted neighborhoods. In [14], several fixed neighborhoods including random neighborhoods have been studied. Dynamic neighborhoods in which the neighborhood of a particle was defined as the k closest individuals in every iteration have been investigated in [15]. The use of such a dynamic neighborhood is computationally intensive because the neighborhood has to be determined anew at every iteration. In [16], the particles have been clustered in every iteration and the centroid of the cluster was used as an attractor instead of using the position of a single individual.

In this paper, we propose a hierarchical version of PSO (H-PSO). In H-PSO, a particle is influenced by its own so far best position and by the best position of the particle that is directly above it in the hierarchy. In H-PSO, all particles are arranged in a tree that forms the hierarchy so that each node of the tree contains exactly one particle. In order to give the best particles in the swarm a high influence, particles move up and down the hierarchy. If a particle at a child node has found a solution that is better than the best so far solution of the particle at the parent node, the two particles are exchanged. We also introduce variants of H-PSO in which the structure of the hierarchy is dynamically changed in order to further improve the search success. Moreover, variants of H-PSO are described in which the behavior of a particle is determined by its position in the hierarchy.

The paper is organized as follows. The PSO method is explained in Section II. The hierarchical PSO algorithm and its variants are described in Section III. In Section IV, the setup for the experiments is described and in Section V, the results are presented and discussed. A conclusion is given in Section VI.

II. PSO

In this section, we describe the PSO method for function optimization (see also [1]). PSO is an iterative method that is based on the search behavior of a swarm of m particles in a multidimensional search space. In each iteration the velocities and positions of all the particles are updated. For each particle i, its velocity vector $\overrightarrow{v_i}$ is updated according to (1). The inertia

Manuscript received September 7, 2004; revised December 21, 2004 and January 28, 2005. This work was supported by the German Research Foundation (DFG) through the project "Swarm Intelligence on Reconfigurable Architectures". This paper was recommended by Associate Editor M. Dorigo.

The authors are with the Parallel Computing and Complex Systems Group, Department of Computer Science, University of Leipzig, D-04109 Leipzig, Germany (e-mail: janson@informatik.uni-leipzig.de; middendorf@informatik.uni-leipzig.de).

Digital Object Identifier 10.1109/TSMCB.2005.850530

weight $w > 0$ controls the influence of the previous velocity vector. The current position of the particle is denoted by $\overrightarrow{x_i}$. Parameter $c_1 > 0$ controls the impact of the personal best position $\overrightarrow{y_i}$, i.e., the position where the particle found the smallest function value so far—assuming that the objective function has to be minimized. Parameter $c_2 > 0$ determines the impact of the best position $\overrightarrow{\hat{y}_i}$ that has been found so far by any of the particles in the neighborhood of particle i. Usually c_1 and c_2 are set to the same value. Random values r_1 and r_2 are drawn with uniform probability from [0,1] for each particle at every iteration.

After all the particles' velocities have been updated, the particles move with their new velocity to their new positions (2). Then, for each particle i the objective function f is evaluated at its new position. If $f(\overrightarrow{x_i}(t+1)) < f(\overrightarrow{y_i})$ the personal best position $\overrightarrow{y_i}$ is updated accordingly, i.e., $\overrightarrow{y_i}$ is set to $\overrightarrow{x_i}(t+1)$

$$\overrightarrow{v_i}(t+1) = w \cdot \overrightarrow{v_i}(t) + c_1 \cdot r_1 \cdot (\overrightarrow{y_i} - \overrightarrow{x_i}) + c_2 \cdot r_2 \cdot (\overrightarrow{\hat{y}_i} - \overrightarrow{x_i}) \quad (1)$$

$$\overrightarrow{x_i}(t+1) = \overrightarrow{x_i}(t) + \overrightarrow{v_i}(t+1). \quad (2)$$

Several variations of this basic PSO scheme have been proposed in the literature. Commonly used variations are to restrict the velocity of a particle by a maximal value $v_{\max}$ or to linearly decrease w over time [17]. This is done to adjust the swarm's behavior from exploration of the entire search space to exploitation of promising regions.

Various mechanisms have been designed to increase the diversity among the particles of a swarm. In [18] a spatial extension is assigned to the particles and different collision strategies are used to avoid crowding of the swarm. A charged swarm (CPSO) is proposed in [19], where some or all the particles hold a certain charge and a repulsion force is applied if two particles get too close to each other. The ARPSO algorithm [20] switches between phases of attraction and repulsion. If the swarm becomes too tight, i.e., the diversity diminishes, the repulsion phase is initiated and the swarm is scattered. In [21], a predator particle is introduced that pursues the global best particle and thereby chases other particles away.

A. Neighborhood Topologies

Different neighborhood topologies have been investigated for PSO. In the original PSO algorithm—here called the *gbest* model—the swarm is guided by the current global best particle, i.e., $\overrightarrow{\hat{y}_i}$ in (1) is the best solution found so far by the swarm. The *gbest* model corresponds to a fully connected neighborhood. In [22], other neighborhood topologies, varying the degree of interconnections between the particles, have been introduced.

In this paper, we also consider the *lbest* model, that uses the local neighborhood best position to update a particle's velocity. The local neighborhood is defined by a ring topology through the particle's index, so that particle i is neighbored to particles $i+1(\bmod m)$ and $i-1(\bmod m)$, where m is the total number of particles. It has been shown [22] that the relative performance of *gbest* and *lbest* depends on the type of the optimization function. In general, *gbest* performs better on unimodal functions, as Sphere and Rosenbrock, whereas *lbest* is better suited for multimodal functions, in which the optimization success relies on the diversity of the algorithm to not being trapped in a local minimum.

In [15], a neighborhood scheme has been explored that is defined by a particle's actual position in the search space. A certain number of close particles are considered to be neighbors. This method is computationally intensive, since in every iteration the distances between all pairs of particles have to be calculated.

A fitness-distance-ratio PSO (FDR-PSO) has been proposed in [23], in which each particle is not only influenced by the personal and global best position but also by a close and good neighbor. This neighbor is selected as the particle that maximizes the quotient of fitness improvement over the respective distance. Thus, any nearby improving neighbor of a particle can be preferred to the global best particle, provided it is close enough.

In [14] and [24], several neighborhood topologies or "sociometries" have been examined for the PSO algorithm. In [24] these topologies have also been applied to the fully informed PSO, in which each particle is influenced by all of its neighbors and not only its best neighbor. The information flow within the swarm is controlled by the two parameters k and C, where k gives the number of neighbors of a specific node in the neighborhood graph and C is used to measure the clustering among the nodes in the neighborhood graph, i.e., to what extent the respective neighborhoods differ. The previously introduced neighborhoods *gbest* and *lbest*, the *star*—one central node is connected to all the other nodes—and other regular neighborhood graphs have been compared to randomly created neighborhood graphs with different values for k and C. The von Neumann neighborhood on a two-dimensional (2-D) lattice performed very good and also a three-dimensional (3-D) pyramid did perform reasonably well. The common *gbest*, *lbest* and *star* topologies all have been rated worse.

III. Hierarchical PSO

The hierarchical version of PSO (H-PSO) is introduced in this section. In H-PSO, all particles are arranged in a hierarchy that defines the neighborhood structure. Each particle is neighbored to itself and its parent in the hierarchy. In this paper, we study regular tree like hierarchies, i.e., the underlying topology is a (nearly) regular tree. The hierarchy is defined by the *height* h, the *branching degree* d, i.e., the maximum number of children of the inner nodes, and the *total number of nodes* m of the corresponding tree (for an example, see Fig. 1). In this paper we use only hierarchies in which all inner nodes have the same number of children, only the inner nodes on the deepest level might have a smaller number of children so that the maximum difference between the number of children of inner nodes on the deepest level is at most one.

In order to give the best individuals in the swarm a high influence, particles move up and down the hierarchy. In every iteration, after the evaluations of the objective function at the particles actual positions, but before the update of the velocities and the determination of the new positions in the search space, the new positions of the particles within the hierarchy are determined as follows. For every particle j in a node of the tree, its own best solution is compared to the best solution found by

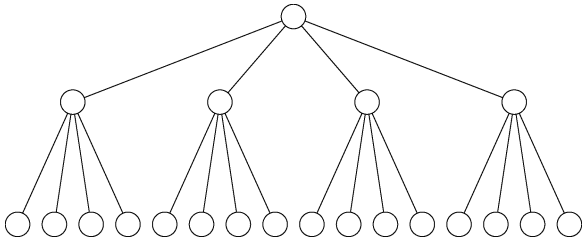

Fig. 1. Example of a hierarchy defined by a regular tree with $h = 3$, $d = 4$, $m = 21$.

the particles in the child nodes. If the best of these particles, say particle i, is better (i.e., $f(\overrightarrow{y_i}) < f(\overrightarrow{y_j})$) then particles i and j swap their places within the hierarchy. These comparisons are performed starting from the top of the hierarchy and then proceed in a breadth-first manner down the tree. Observe that the top-down approach implies that in one iteration an individual can move down several levels in the hierarchy but it can move up at most one level. The current global best particle will move up one level of the hierarchy at every iteration. Hence, it will be on top of the hierarchy after at most $h - 1$ iterations—unless, a better solution was found meanwhile.

For the update of the velocities in H-PSO, a particle is influenced by its own so far best position and by the best position of the individual that is directly above in the hierarchy. This means that for particle i the value of $\overrightarrow{\hat{y}_i}$ in (1) equals $\overrightarrow{y_j}$ where j is the particle in the parent node of particle i. Only when particle i is in the root of the tree, H-PSO uses $\overrightarrow{\hat{y}_i} = \overrightarrow{y_i}$.

Similar as in PSO, after the particles' velocities are updated and after the particles have moved in H-PSO, the objective function is evaluated at the new position. If the function value at this position is better than the function value at the personal best position, the new position is stored in $\overrightarrow{y_i}$.

A. Neighborhood

We propose a new neighborhood scheme for PSO that uses the particle's so far best found function value to define the neighborhood relations. This approach is similar to the *lbest* model in the fact that only a certain fraction of the swarm is considered for the velocity update of a particle. But in our algorithm the neighborhoods are constantly changing, according to the fitness development of the individuals.

The changing arrangement of the particles can help preserving diversity in the search. In the described hierarchy the arrangement of the particles leads to a different influence for the particles at different positions. The particle with the currently best found solution can (indirectly) influence all the other particles after it has reached the top of the hierarchy. This characteristic of H-PSO is similar to the *gbest* model.

B. Adapting the Hierarchy

It is to be expected that the structure of the hierarchy, i.e., the branching degree d, has a significant influence on the optimization behavior of H-PSO. For example, H-PSO with a high branching degree d might perform better in the beginning of the optimization process because all particles are close to the top particle in the hierarchy. Moreover, this tree topology has a small diameter similar to the *gbest* neighborhood, that, in general, optimizes faster than the PSO algorithm using the *lbest* neighborhood. On the other hand, the quality increase of the best solutions found by H-PSO with a smaller d might be slower in the beginning of the optimization process but it might improve the objective function value further in the end of the optimization process (see Section V-A).

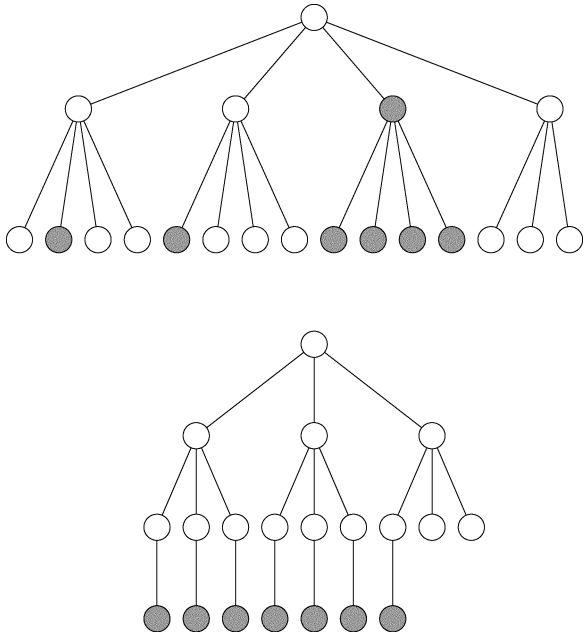

Fig. 2. Example for adapting the hierarchy of AH-PSO ($m = 20$) from $d = 4$ to $d = 3$.

These expectations have inspired the idea to dynamically change the branching degree of the hierarchy during a run. A related idea has been used in [15] where the neighborhood of the particles was gradually increased from *lbest* to *gbest*. Surprisingly, the opposite direction, which might be more effective, has not been tested. In the Adaptive H-PSO (AH-PSO) algorithm that is proposed here, the branching degree is gradually decreased during a run of the algorithm. In order to decrease the branching degree from d to $d - 1$, the hierarchy is traversed starting at the root node. This is done so that always one of the direct subtrees below the considered node is removed, if the number of children exceeds the new required branching degree (see Fig. 2 and the next paragraph for an example). The decision about which subtree to remove is based on the quality of the particles in the topmost nodes of all subtrees of the considered nodes, i.e., all children of the considered node. This procedure is repeated for the entire tree. After this first phase of the branching degree reduction procedure the remaining tree is of branching degree $d - 1$ but has fewer nodes than before. The removed nodes are then evenly inserted at the bottom of the hierarchy. Starting with the node on the second to last level which has the least number of successors. The removed nodes are appended one by one so that the number of children of all nodes on the second last level differ by at most one. If all of these nodes have $d - 1$ children, a new level is added to the hierarchy and the procedure is continued until all removed nodes are reinserted.

As an example, consider Fig. 2, where a hierarchy of 20 nodes and branching degree $d = 4$ is shown before and after the branching degree has been reduced to $d = 3$. The grey nodes are those that have first been removed and then appended at the bottom level. Note, that none of the leaves of the right subtree have been removed since their parent node already has branching degree 3.

In the test runs that have been done for this paper, the branching degree reduction operation is performed every f_{adapt}-th iteration of AH-PSO. Parameter f_{adapt} is called the decrease frequency. At every branching degree reduction operation the branching degree is decreased by k_{adapt}. This parameter is called decrease step size. For $k_{\text{adapt}} > 1$ the reduction procedure is applied consecutively (i.e., the branching degree is always reduced in steps of 1) until the hierarchy has the required branching degree. This is done until a certain minimum branching degree $d_{\min}$ is reached.

In order to decrease the branching degree of a node, it has to be decided which of its direct subtrees has to be removed. In preliminary experiments, we tested two strategies: removing the subtree with the worst root node or the one with the best root node. Since removing the best successor always provided better results in these preliminary tests, we use only this strategy for the experiments that are described in this paper. A possible explanation why it is worse to remove the subtree with the worst root could be that in this case the removed particles that are appended to the bottom of the hierarchy have only small chances to ascend in the hierarchy. Therefore, it is likely that they are not immediately useful for the further optimization process. The particles in the subtree with the best root node on the other hand are holding good personal best positions already and can thus immediately contribute to the collective search process.

C. Specialization

A common feature of self-organizing swarm behavior in nature is the specialization of certain individuals to certain tasks. In such systems, the selection of which task an individual is working on is usually modeled to be triggered by an external stimulus. The stimulus is typically filtered by a threshold value that depends on the current state of the individual. In [25], a Division-of-Labor-PSO algorithm has been presented that identifies two different tasks for the individuals: the regular exploration task and a so called local search task. In the latter task, a particle is placed on the current global best position with a new random velocity vector. The general idea is that if a particle has not improved its fitness at an iteration, the threshold for engaging in the local search task gradually decreases.

The H-PSO tree topology facilitates the assignment of different tasks to the individuals. The particles are roughly ordered by fitness into the different levels of the hierarchy. This does reflect a current, relative state of the individuals and can thus be used as a distinction for assigning different behaviors to the particles. Thus, instead of using a threshold based division of labor approach, we associate various tasks to the particles in different levels of the hierarchy.

It is a common modification of the basic PSO algorithm to linearly decrease the value of parameter w over time [17]. This is done to adjust the behavior of the swarm from exploration of the entire search space to exploitation of promising regions. We identify different search tasks by different inertia weights w_k. Each level k of the hierarchy is assigned a certain weight w_k and all particles in that level behave accordingly. The values w_k are determined using either (3) or (4), with level $k \in [0, h-1]$ (the root is on level 0) and the resulting $w_k \in [w_{\min}; w_{\max}]$. The algorithm using (3) has the w_k values decreasing, from bottom to top of the hierarchy, with the root particle using $w_{\min}$. This algorithm is denoted $\bigwedge \mathrm{H-PSO}$. The algorithm using (4) inverts this assignment with the root particle using $w_{\max}$ and it is denoted $\bigvee \mathrm{H-PSO}$

$$w_k = \frac{(w_{\max} - w_{\min}) \cdot k}{h-1} + w_{\min} \tag{3}$$

$$w_k = \frac{(w_{\min} - w_{\max}) \cdot k}{h-1} + w_{\max}. \tag{4}$$

Different behaviors, controlled by different values of the inertia weight w for each particle, have also been used by other authors before. In [26], the PSO algorithm has been combined with the concept of self-organized criticality. Each particle holds a current critical value that increases when the neighboring particles get closer. If enough criticality is accumulated, this criticality is dispersed, thereby maybe exceeding the criticality limit of other particles, and a relocation scheme is initiated. Possible

TABLE I
TEST FUNCTIONS

Sphere:

$$F_{Sph}(\vec{x}) = \sum_{i=1}^{n} x_i^2$$

Rosenbrock:

$$F_{Ros}(\vec{x}) = \sum_{i=1}^{n-1} \left(100\,(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right)$$

Rastrigin:

$$F_{Ras}(\vec{x}) = \sum_{i=1}^{n} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right)$$

Griewank:

$$F_{Gri}(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

Schaffer's f6:

$$F_{Sch}(\vec{x}) = 0.5 - \frac{\left(\sin\sqrt{x_1^2 + x_2^2}\right)^2 - 0.5}{\left(1 + 0.001(x_1^2 + x_2^2)\right)^2}$$

Ackley:

$$F_{Ack}(\vec{x}) = -20 \cdot \exp\left(-0.2\ \sqrt{1/n \cdot \sum_{i=1}^{n} x_i^2}\right) - \exp\left(1/n \cdot \sum_{i=1}^{n} \cos(2\pi \cdot x_i)\right) + 20 + e$$

TABLE II
PARAMETERS FOR TEST FUNCTIONS

Name	Dim	Initial Range	Goal
Sphere	30	$[-100; 100]^n$	0.01
Rosenbrock	30	$[-30; 30]^n$	100
Rastrigin	30	$[-5.12; 5.12]^n$	100
Griewank	30	$[-600; 600]^n$	0.1
Schaffer's f6	2	$[-100; 100]^2$	10^{-5}
Ackley	30	$[-32; 32]^n$	0.1

relocation schemes include resetting the personal best positions or pushing the particle forward in its current direction of movement. Another approach varied the inertia weight w according to the current criticality level of a particle.

IV. EXPERIMENTS

In this section, the experiments that have been done to compare the different variants of PSO for continuous function optimization are described. Since, in this paper, we are interested in understanding whether the proposed modifications of the standard PSO algorithm can improve its performance, we focus our experimental evaluation on the comparison with other PSO algorithms. It should be noted however that PSO is known to be a competitive method which often produces results that are comparable or even better than those produced by other metaheuristics (e.g., see [27]). In all our experiments, the PSO algorithms use the parameter values $w = 0.729$ and $c_1 = c_2 = 1.494$ as recommended in [28], unless stated otherwise. Each run has been repeated 100 times and average results are presented. The particles have been initialized with a random position and a random velocity where in both cases the values in every dimension have been randomly chosen according to a uniform distribution over the initial range $[X_{\min}; X_{\max}]$. The values $X_{\min}$ and $X_{\max}$ depend on the objective function. During a run of an algorithm, the position and velocity of a particle have not been restricted to the initialization intervals, but a maximum velocity $v_{\max} = X_{\max}$ has been used for every component d of velocity vector $\overrightarrow{v_i}$.

The set of test functions (see Table I) contains functions that are commonly used in the field of continuous function optimization. Table II shows the values that have been used for the dimension of these functions, the range of the corresponding initial position and velocities of the particles, and the goals that have to be achieved by the algorithms. The first two functions (Sphere and Rosenbrock) are unimodal functions (i.e., they have a single local optimum that is also the global optimum) and the remaining four functions are multimodal (i.e., they have several local optima). All test runs have been run over 10 000 iterations.

When comparing different hierarchies for H-PSO, the swarm size has been kept constant for all different values of $d \in \{2, \ldots, 20\}$. The resulting hierarchies can then become irregular. As was described before, we appended the additional (with respect to the largest regular arrangement) nodes evenly below the last level. For example, the regular H-PSO topology with $d = 4,\ h = 3$ has a resulting size of $m = 21$. For a swarm of size $m = 40$ the remaining 19 nodes are appended to the 16 nodes at the bottom level. Hence, each node receives one successor and three nodes receive a second successor.

The H-PSO and AH-PSO algorithms have been compared to the PSO algorithm using the *gbest* (PSO-g) and the *lbest* (PSO-l) neighborhoods. The swarm size that has been used in the experiments is $m = 40$. For H-PSO, the parameter values $d = 5$ and $h = 3$ have been used. The branching degree of AH-PSO started with $d = 20$ and has been decreased until $d_{\min}$. For the multimodal test functions, $d_{\min}$ has been set to the respective optimum branching degree for each test function that was determined by comparing different values of d. The branching degree is decreased every f_{adapt} iterations by k_{adapt} steps. Several combinations of values for these parameters have been tested, but only the results for the combination with the best values are reported for each function. The tested values were all combinations of decrease step size $k_{\text{adapt}} \in \{1, 2, 4\}$ and decrease frequency $f_{\text{adapt}} \in \{5, 10, 50, 100, 500, 1000\}$.

In another experiment the number of iterations required to reach a certain goal for each test function has been determined comparing PSO-g, PSO-l, H-PSO, $\bigwedge \mathrm{H-PSO}$ and $\bigvee \mathrm{H-PSO}$. The parameters values that have been used in this experiment are $m = 31,\ h = 3$, and $d = 5$. A swarm size of $m = 31$ has been used in order to obtain a regular hierarchy for the H-PSO variants and to stay close to the common swarm size of $m = 30$ that has been used, for example, in [28]. For $\bigwedge \mathrm{H-PSO}$, the values of w_k in the hierarchy varies from $w_0 = 0.4$ to $w_2 = 0.729$. Vice versa for $\bigvee \mathrm{H-PSO}$ the parameter values vary from $w_0 = 0.729$ to $w_2 = 0.4$.[1] For PSO-g, PSO-l and H-PSO 2 parameter sets have been used. All algorithms have been tested with two different parameter sets that were taken from the literature.

One parameter set is $w = 0.6$ and $c_1 = c_2 = 1.7$ as suggested in [28] for a faster convergence rate. The other parameter set has the common parameter values $w = 0.729$ and $c_1 = c_2 = 1.494$. Algorithms that use the first (second) set of parameter values are denoted by appending "−a" (respectively "−b") to the name, e.g. PSO-g-a denotes PSO-g with the first parameter set.

A. Significance

In the experiments, the number of iterations required to reach a specified goal for each function was recorded for each algorithm. If the goal was not reached within the maximum number of 10 000 iterations, the run was considered unsuccessful. The success rate denotes the percentage of successful runs. For the successful runs, the average, median, maximum, and minimum number of iterations required to achieve the goal value was calculated. The expected number of iterations has been determined as (average/success rate). We also evaluated the significance of observed differences between the algorithms.

[1]We also tried PSO-g, PSO-l and H-PSO with $w = 0.4$ which did not produce competitive results.

TABLE III
STEPS REQUIRED TO ACHIEVE A CERTAIN GOAL FOR PSO-g, PSO-l, H-PSO, $\bigwedge$H-PSO AND $\bigvee$H-PSO. AVERAGE (AVG), MEDIAN (MED), MAXIMUM (MAX), MINIMUM (MIN) AND EXPECTED (Exp := Avg/Succ) NUMBER OF ITERATIONS AND SIGNIFICANCE MATRIX; "−a" AND "−b" DENOTE THE USED PARAMETER VALUES AS DESCRIBED IN SECTION IV

Algorithm		Avg	Med	Max	Min	Suc	Exp	S-Matrix
		Sphere						1 2 3 4 5 6 7 8
PSO-g-a	(1)	309.4	303.0	530.0	238.0	1.00	309.4	X X X X X X -
PSO-l-a	(2)	449.4	452.0	482.0	406.0	1.00	449.4	- - - - X - - -
H-PSO-a	(3)	360.0	361.0	414.0	301.0	1.00	360.0	- X - - X X - -
PSO-g-b	(4)	363.0	355.0	539.0	289.0	1.00	363.0	- X - - X X - -
PSO-l-b	(5)	563.2	563.0	625.0	516.0	1.00	563.2	- - - - - - - -
H-PSO-b	(6)	453.9	453.0	529.0	388.0	1.00	453.9	- - - - X - - -
$\bigwedge$H-PSO	(7)	351.5	348.0	503.0	301.0	1.00	351.5	- X X - X X - -
$\bigvee$H-PSO	(8)	**209.6**	**205.0**	**319.0**	**173.0**	1.00	**209.6**	X X X X X X X
		Rastrigin						1 2 3 4 5 6 7 8
PSO-g-a	(1)	**104.0**	**101.5**	**191.0**	64.0	0.98	**106.1**	X X X X X X X X
PSO-l-a	(2)	185.7	176.0	611.0	102.0	0.99	187.6	- X - X X - - -
H-PSO-a	(3)	432.7	291.0	2482.0	95.0	0.99	437.1	- - - - X - - -
PSO-g-b	(4)	142.0	132.0	282.0	82.0	0.98	144.9	- X X - X X - -
PSO-l-b	(5)	270.0	220.5	3657.0	119.0	1.00	270.0	- - X - - X - -
H-PSO-b	(6)	500.9	367.5	1703.0	142.0	1.00	500.9	- - - - - - - -
$\bigwedge$H-PSO	(7)	151.2	127.0	677.0	65.0	1.00	151.2	- X X - X X - -
$\bigvee$H-PSO	(8)	184.4	117.0	4192.0	**52.0**	1.00	184.4	- X X X X X -
		Rosenbrock						1 2 3 4 5 6 7 8
PSO-g-a	(1)	497.1	302.5	4189.0	195.0	1.00	497.1	X X X X X X X -
PSO-l-a	(2)	704.3	497.0	7851.0	326.0	1.00	704.3	- - - X - - - -
H-PSO-a	(3)	528.3	340.5	5924.0	247.0	1.00	528.3	- X - - X X - -
PSO-g-b	(4)	641.2	382.5	5165.0	230.0	1.00	641.2	- X - - X X - -
PSO-l-b	(5)	798.0	615.0	5164.0	408.0	1.00	798.0	- - - - - - - -
H-PSO-b	(6)	780.2	471.0	5315.0	295.0	1.00	780.2	- - - - X - - -
$\bigwedge$H-PSO	(7)	702.0	369.5	4183.0	238.0	1.00	702.0	- X - - X X - -
$\bigvee$H-PSO	(8)	**352.7**	**262.5**	**2251.0**	**144.0**	1.00	**352.7**	X X X X X X X
		Schaffer						1 2 3 4 5 6 7 8
PSO-g-a	(1)	572.0	**126.0**	8335.0	52.0	0.70	817.1	X X - X X X -
PSO-l-a	(2)	905.1	432.5	7610.0	**45.0**	0.98	923.6	- - - - - - - -
H-PSO-a	(3)	436.6	249.0	4023.0	72.0	1.00	436.6	- X - - X - - -
PSO-g-b	(4)	956.1	179.0	8866.0	55.0	0.70	1365.8	- X - - X - - -
PSO-l-b	(5)	842.9	446.0	9293.0	73.0	0.99	851.4	- - - - - - - -
H-PSO-b	(6)	389.9	208.0	3508.0	54.0	1.00	389.9	- X - - X - - -
$\bigwedge$H-PSO	(7)	646.3	284.0	7903.0	**45.0**	0.99	652.8	- X - - X - - -
$\bigvee$H-PSO	(8)	**317.2**	201.0	**2466.0**	55.0	0.97	**327.0**	- X - - X - -
		Griewank						1 2 3 4 5 6 7 8
PSO-g-a	(1)	263.9	260.0	368.0	215.0	0.95	277.8	X X X X X X X -
PSO-l-a	(2)	416.7	413.0	501.0	343.0	1.00	416.7	- - - - X - - -
H-PSO-a	(3)	324.9	320.0	430.0	271.0	1.00	324.9	- X - - X X - -
PSO-g-b	(4)	322.0	311.0	672.0	248.0	0.96	335.4	- X - - X X - -
PSO-l-b	(5)	515.0	512.5	638.0	451.0	1.00	515.0	- - - - - - - -
H-PSO-b	(6)	410.1	411.0	608.0	338.0	0.99	414.2	- - - - X - - -
$\bigwedge$H-PSO	(7)	307.7	308.0	382.0	237.0	0.92	334.5	- X X - X X - -
$\bigvee$H-PSO	(8)	**184.6**	**183.0**	**278.0**	**147.0**	0.95	**194.3**	X X X X X X X
		Ackley						1 2 3 4 5 6 7 8
PSO-g-a*	(1)	218.0	218.0	218.0	218.0	0.01	21800.0	- - - - - - - -
PSO-l-a	(2)	422.3	415.0	537.0	356.0	1.00	422.3	- - - - X - - -
H-PSO-a	(3)	324.1	323.5	406.0	261.0	0.86	376.9	- X - - X X - -
PSO-g-b	(4)	319.0	319.0	327.0	311.0	0.03	10633.3	- X - - X X - -
PSO-l-b	(5)	522.9	519.0	672.0	442.0	0.99	528.2	- - - - - - - -
H-PSO-b	(6)	417.0	416.0	555.0	334.0	0.87	479.3	- - - - X - - -
$\bigwedge$H-PSO†	(7)	346.8	341.0	451.0	268.0	0.11	3152.9	- X - - X X - -
$\bigvee$H-PSO†	(8)	**225.9**	**226.0**	**273.0**	**194.0**	0.98	**230.5**	- X X X X X X

*Excluded from the comparison, because the values are based on only one successful run.

†Branch degree $d = 2$ was used for these experiments, because $\bigwedge$H-PSO and $\bigvee$H-PSO with degree $d = 5$ both achieved only a poor success rate.

For evaluating the significance the Wilcoxon Rank Sum Test has been used to compare the results for two algorithms. The results of the 100 test runs for two algorithms form two independent samples but only the successful runs are considered, thus the used sample sizes can differ. For two algorithms (X and Y) the distribution of their results, F_X and F_Y, are compared using

the null-hypothesis H_0: $F_X = F_Y$ and the one-sided alternative H_1: $F_X < F_Y$. We performed the tests at a significance level of $\alpha = 0.01$. In the results section the significance comparison among a set of k algorithms is displayed using a $k \times k$ matrix $A = [a_{ij}]_{i,j\in[1:k]}$, in which an entry "X" at position $a_{i,j}$ denotes that algorithm i is significantly better, i.e., it provides smaller values than algorithm j (where i and j are the position in the corresponding result table). For example, the leftmost X in the first row of Table III indicates that algorithm PSO-g-a is significantly better than algorithm PSO-l-a. An entry "-" at position $a_{i,j}$ indicates that algorithm i is not significantly better than algorithm j. The entries $a_{i,i}$ on the main diagonal are left empty. We call the corresponding matrix a significance matrix (s-matrix).

B. Diversity

The hierarchy of H-PSO divides the swarm into several subswarms of particles that are located in different subtrees. This division is not persistent, as the particles can move between these subswarms. We study in this paper whether the different subtrees in a H-PSO hierarchy can still specialize to different regions of the search space as intended by the design of H-PSO. Therefore, the diversity of the particles in the subtrees has been measured. The corresponding test runs were done with parameter values $m = 31$, $h = 3$, and $d = 5$. The resulting topology consists of five subtrees below the root node, each subtree containing six nodes. The diversity has been measured for each of these five subtrees. Since the size of a particle set influences the diversity values, we can not directly compare the diversity of the particles within a subtree with the diversity of the whole swarm. Therefore, the diversity within a subtree is also compared to the diversity of a subset of six particles that have been selected randomly with uniform probability from the whole swarm.

Determining the diversity within the swarm is based on the "distance-to-average-point" measure δ_{avgPoint} given by (5), where S is a subset of the swarm and $|S|$ the size of S. The problem dimension is denoted by n and x_{id} is the d-th component of $\overrightarrow{x_i}$. The average point of the particles in S at iteration t is given as $\overrightarrow{\overline{x}}(t)$

$$\delta_{\text{avgPoint}}(S,t) = \frac{1}{|S|} \cdot \sum_{i\in S} \sqrt{\sum_{d=1}^{n} (x_{id}(t) - \overline{x}_d(t))^2}. \quad (5)$$

The "distance-to-average-point" measure returns an unbounded, absolute value that depends on the considered test function and the current diameter of the swarm. For comparison we want to use a measure that is independent of the test function and the state of the optimization process. Therefore, δ_{avgPoint} is scaled by the diameter of the swarm. The diameter is determined as the maximum distance $\delta_{\max}$ between any two particles in the entire swarm (not just particles from subset S)—see (6)

$$\delta_{\max}(t) = \max_{i,j\in\{1,\ldots,n\}} \left\{ \sqrt{\sum_{d=1}^{n} (x_{id}(t) - x_{jd}(t))^2} \right\}. \quad (6)$$

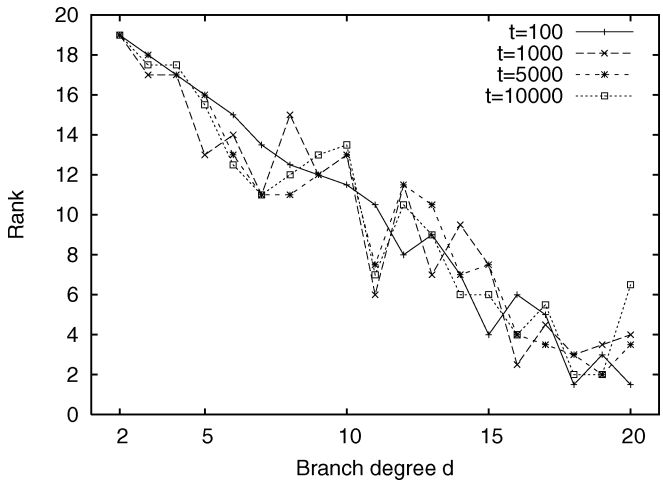

Fig. 3. Average ranks for unimodal functions for H-PSO for different branch levels ($m = 40$).

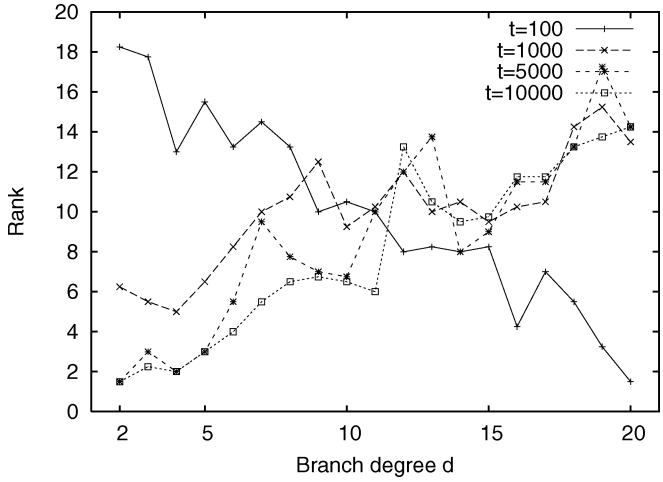

Fig. 4. Average ranks for multimodal functions for H-PSO for different branch levels ($m = 40$).

As diversity measure $D(S,t)$ of a subset S of the swarm at iteration t measure $D(S,t) = \delta_{\text{avgPoint}}(S,t)/\delta_{\max}(t)$ is used. Scaling $D(S,t)$ by $\delta_{\max}(t)$ preserves the relations between different subsets, since all measures are scaled by the same value.

V. Results

A. Branch Degree

The results of the experiment in which the influence of the branch degree of the H-PSO hierarchy on the optimization behavior has been investigated are shown in Fig. 3 (for the unimodal test functions) and in Fig. 4 (for the multimodal test functions). The branch degree d varies over all values $\{2,\ldots,20\}$. For every test function the solution qualities that were achieved by the algorithms using the different parameter settings have been ranked at each iteration and for each value of $d \in \{2,\ldots,20\}$ the average rank (over the unimodal respectively multimodal test functions) is shown. In the figures the ranks for the different branch degrees are displayed for iterations $t \in \{100, 1000, 5000, 1000\}$.

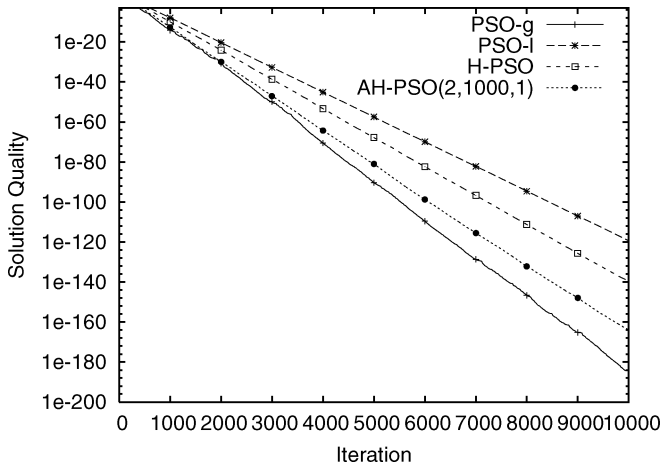

Fig. 5. Sphere—solution quality for PSO-g, PSO-l, H-PSO ($h = 3, d = 5$) and AH-PSO; swarm size $m = 40$.

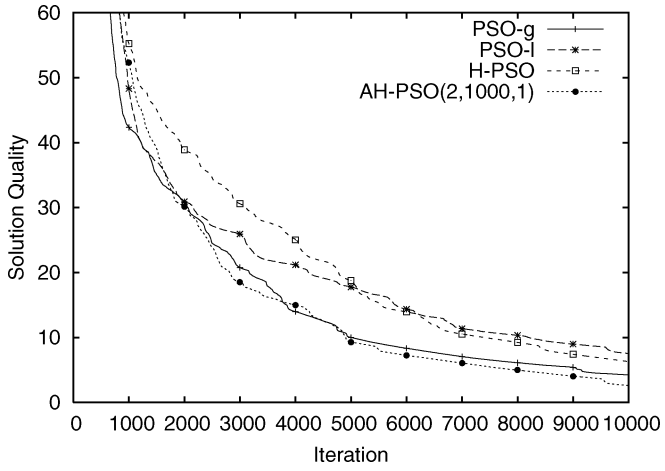

Fig. 6. Rosenbrock—solution quality for PSO-g, PSO-l, H-PSO ($h = 3, d = 5$) and AH-PSO; swarm size $m = 40$.

The results for the two unimodal test functions show that higher branch degrees lead to a better optimization behavior over the entire duration of the test runs. Only for branching degrees higher than $d \geq 15$, the average ranks do not differ much because for $m = 40$ the hierarchies become very similar. For the multimodal test functions, the results show a clearly different behavior. In the beginning of the optimization process $(t = 100)$ again the H-PSO algorithms with higher branching degrees achieved better ranks, but this tendency is inverted for later iterations $t \in \{1000, 5000, 10000\}$ in which the smaller degrees are better. This shows that a flexible topology in which the branch degree changes during the optimization process might lead to better results for multimodal test functions.

B. Topology

In this subsection, the H-PSO and AH-PSO algorithm (with parameter values $h = 3$, $d = 5$) are compared to PSO using the two neighborhood topologies *gbest* (PSO-g) and *lbest* (PSO-l). The swarm size that has been used is $m = 40$. As shown in the last subsection, the optimization success of a neighborhood topology (*gbest*, *lbest*) highly depends on the kind of function to be optimized (unimodal or multimodal). For Sphere (Fig. 5) and Rosenbrock (Fig. 6) PSO-g performes better than PSO-l. On the other hand, PSO-l achieves better results than PSO-g for Rastrigin (Fig. 7), Griewank (Fig. 8), Schaffer (Fig. 9) and Ackley (Fig. 10).

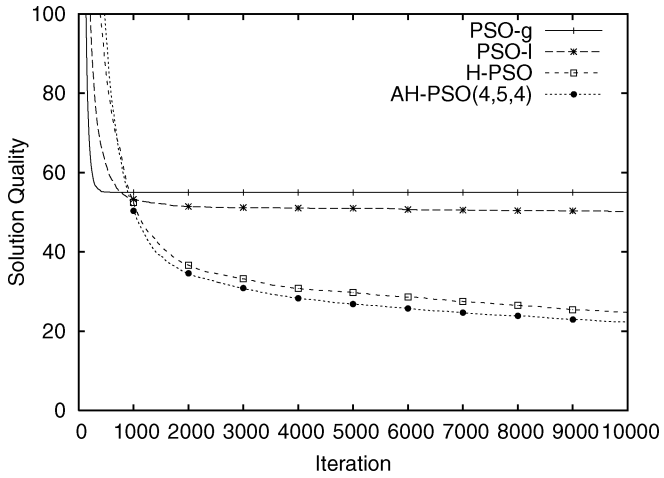

Fig. 7. Rastrigin—solution quality for PSO-g, PSO-l, H-PSO ($h = 3, d = 5$) and AH-PSO; swarm size $m = 40$.

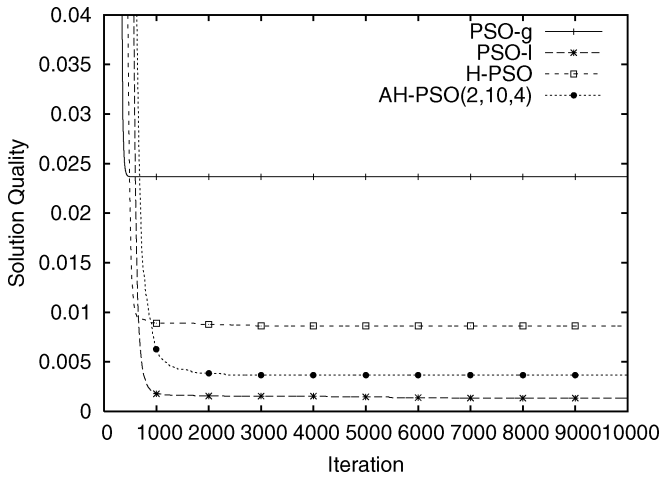

Fig. 8. Griewank—solution quality for PSO-g, PSO-l, H-PSO ($h = 3, d = 5$) and AH-PSO; swarm size $m = 40$.

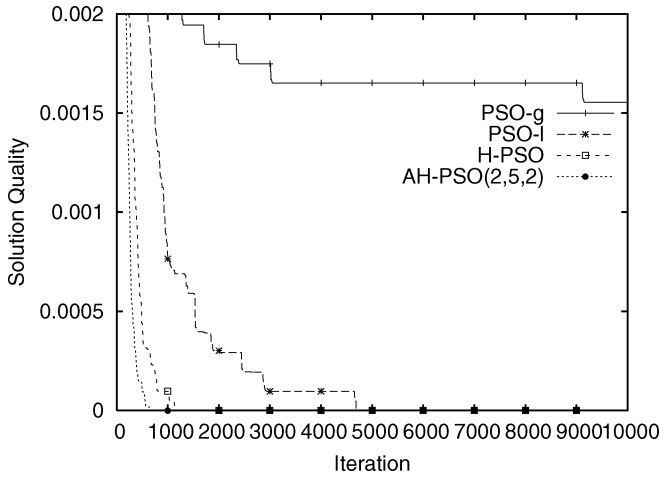

Fig. 9. Schaffer—solution quality for PSO-g, PSO-l, H-PSO ($h = 3, d = 5$) and AH-PSO; swarm size $m = 40$.

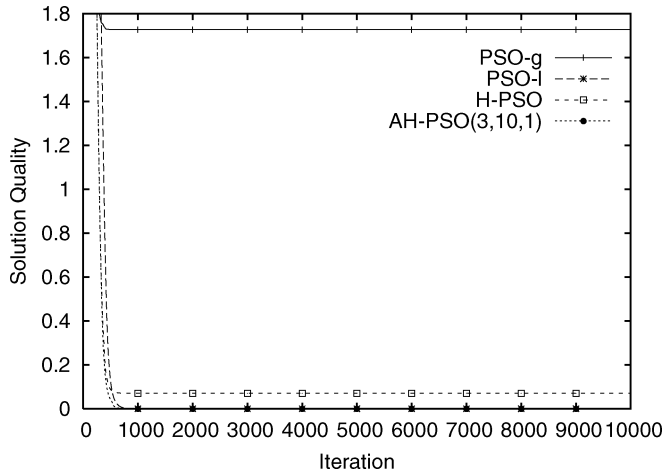

Fig. 10. Ackley—solution quality for PSO-g, PSO-l, H-PSO ($h = 3$, $d = 5$) and AH-PSO; swarm size $m = 40$.

The results show that the optimization behavior of H-PSO does not depend so much on the specific function type. H-PSO produces competitive results for all the considered functions, regardless of whether they are unimodal or multimodal. For all test functions, H-PSO performs clearly better than the worst one of PSO-l and PSO-g. On some test functions (Rastrigin and Schaffer), it performs even better than both PSO versions. It should be noted that these results have been obtained with the same fixed branch degree $d = 5$. Hence, the value $d = 5$ can be used as a recommendation for parameter choice. Nevertheless, it can be expected that the results of H-PSO for a specific test function can in general be improved by adapting the value of d.

Using the adaptive H-PSO (AH-PSO) algorithm the results of H-PSO could be improved. For every test function the minimum branch degree $d_{\min}$ of AH-PSO is set to the optimal branch degree of the respective function (determined by the experiments in Section V-A). The AH-PSO parameters are displayed as AH-PSO($d_{\min}$, f_{adapt}, k_{adapt}) for the minimum branch degree $d_{\min}$, the decrease frequency f_{adapt} and the decrease step size k_{adapt}.

For the unimodal test functions—Rosenbrock and Sphere—the AH-PSO algorithm that decreased the branch degree only every 1000 iterations by 1 performs best. The branch degree is only reduced to $d = 11$ until the end of the algorithm. AH-PSO(2,1000,1) performs better than the best performing H-PSO ($d = 18$) for the Rosenbrock function and very similar to the best H-PSO ($d = 20$) for the Sphere function. For all multimodal test problems, the best performing AH-PSO algorithms reduce the branch degree to the optimum degree very fast. In case the Rastrigin function AH-PSO(4,5,4) reaches the optimum degree $d = 4$ after 20 iterations and it performs better than the best H-PSO ($d = 4$). Also, for the other multimodal test functions, AH-PSO performs better than H-PSO with the optimum branch degree. It can be concluded that a fast adaption of the hierarchy during the initial phase of a run seems to be advantageous for multimodal functions, whereas optimization of unimodal functions can benefit from a slow decrease of d over the entire running time.

C. Specialization

In this section, H-PSO and its weighted variants are compared with PSO-l and PSO-g. The comparison is based on the number of iterations required to reach a certain goal for each of the test functions. The algorithms all used swarm size $m = 31$. In Table III, the average, median, maximum and minimum iterations required to achieve the goal value are shown. Also, the success rate and the expected number of iterations (average/success rate) to reach the goal are given.

$\bigvee$H$-$PSO is always among the fastest algorithms to achieve the desired goal. Only for the Rastrigin function, PSO-g-a does require an average of 104.0 iterations, where $\bigvee$H$-$PSO takes 184.4. For all other test functions, $\bigvee$H$-$PSO obtains the lowest average and expected number of iterations required to reach the goal.

The success rate of $\bigvee$H$-$PSO and $\bigwedge$H$-$PSO is in general very high. An exception is the Ackley function for which $\bigvee$H$-$PSO and $\bigwedge$H$-$PSO (with $d = 5$) achieve success rates of only 0.01 and 0.13, respectively. Also, PSO-g has a very low success rate for this function in contrast to PSO-l which has a high success rate. Therefore, $\bigvee$H$-$PSO and $\bigwedge$H$-$PSO have also been tested for a small branching degree of $d = 2$ (these values are included in the table). For the small branching degree $\bigwedge$H$-$PSO has a very high success rate and finds a solution of the required quality on average in less iterations than the PSO-l algorithms.

The significance comparison shows that only for the Rastrigin function PSO-g-a is significantly better than $\bigvee$H$-$PSO. $\bigvee$H$-$PSO performs better for the Sphere, Rosenbrock and Griewank function and also for the Ackley function, considering that PSO-g-a only reached the goal once in all 100 test runs.

The results show that $\bigvee$H$-$PSO can reach a required goal significantly faster than PSO and H-PSO. This demonstrates that H-PSO is a very promising algorithm. The results also indicate that a heterogenous swarm of particles with different values for w seems advantageous.

In order to explain the different optimization behavior of $\bigwedge$H$-$PSO and $\bigvee$H$-$PSO, recall that the parameter values of w_k are ranging from 0.4 to 0.729. Since the majority of the swarm is located in lower levels of the hierarchy, most of the individuals of $\bigvee$H$-$PSO use $w_k = 0.4$. The average w_k for $\bigwedge$H$-$PSO, with $h = 3$, $d = 5$ and $m = 31$, is 0.692, compared to 0.437 for $\bigvee$H$-$PSO. This smaller impact of the previous velocity $\overrightarrow{v_i}(t)$ compared to the personal best and neighborhood best attractors in $\bigvee$H$-$PSO could explain why $\bigvee$H$-$PSO converges faster than $\bigwedge$H$-$PSO.

Another possible explanation can be based on the investigations of the convergence behavior of a deterministic version of PSO. In [29] it was shown for the deterministic PSO that the update equation is stable only if $0.5 \cdot (c_1 + c_2) - 1 > w$. Thus, for $w = 0.4$ ($w = 0.729$) and $c_1 = c_2 = 1.494$, the particle trajectory would diverge (respectively converge). Hence, w must be sufficiently large for a given value of c_1 and c_2 or the trajectory is not stable. Assuming that the results of the deterministic PSO can be transferred, the particle on top of $\bigwedge$H$-$PSO would not have a smooth trajectory but would jump around more randomly compared to the top particle of $\bigvee$H$-$PSO. But care

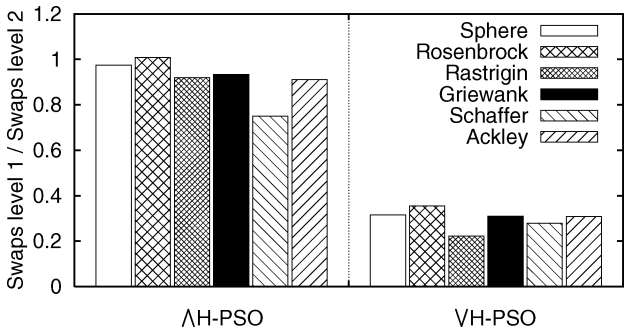

Fig. 11. Ratio of the number of swaps between level 0 and 1 to the number of swaps between level 1 and 2 for $\bigwedge \mathrm{H}-\mathrm{PSO}$ and $\bigvee \mathrm{H}-\mathrm{PSO}$.

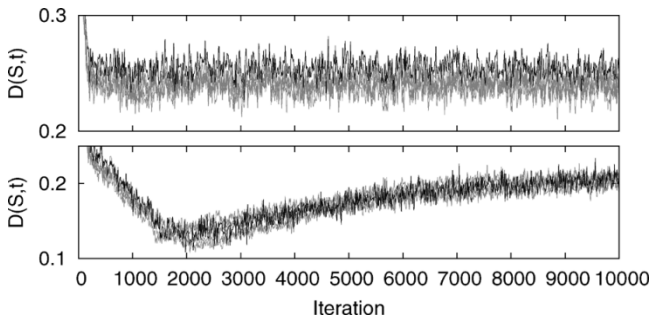

Fig. 12. Diversity $D(S,t)$ for H-PSO with $m = 31$, $h = 3$, $d = 5$ of the five subtrees (grey) and random subset of 13 nodes (black) for the unimodal functions Sphere, Rosenbrock (top to bottom).

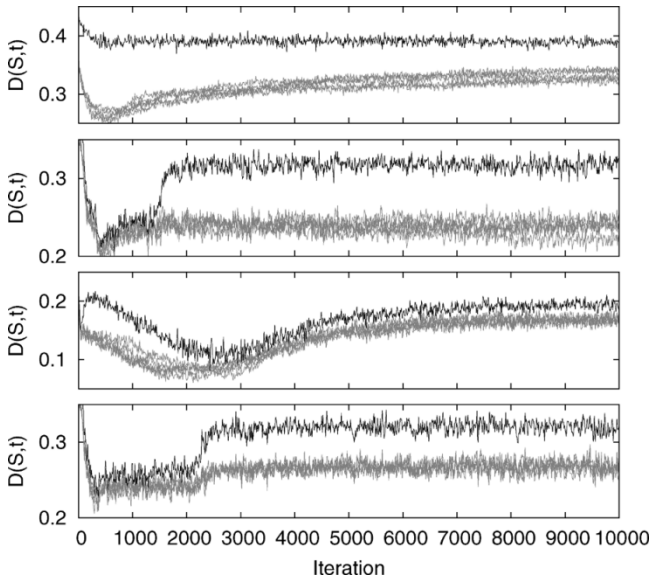

Fig. 13. Diversity $D(S,t)$ for H-PSO with $m = 31$, $h = 3$, $d = 5$ of the five subtrees (grey) and random subset of 13 nodes (black) for the multimodal functions Rastrigin, Griewank, Schaffer, Ackley (top to bottom).

has to be taken with such an explanation since the stochastic PSO variants use $c_1 \cdot r_1$ and $c_2 \cdot r_2$ with an expected value of $1.494 \cdot 0.5 = 0.747$, for $c_1 = c_2 = 1.494$ and r_1, r_2 randomly chosen from [0;1]. For $c_1 = c_2 = 0.747$ the deterministic PSO would converge. Our observations have shown that for the test functions a more likely explanation is that the top particle in $\bigwedge \mathrm{H}-\mathrm{PSO}$ tends to converge too early to the actual point of attraction and is then replaced by a better individual (this might also explain the higher number of swaps for $\bigwedge \mathrm{H}-\mathrm{PSO}$).

During a run of H-PSO, the individuals at different levels of the hierarchy are swapped according to their current fitness. In $\bigwedge \mathrm{H}-\mathrm{PSO}$, the root particle uses $w_0 = 0.4$ and is thus slowed down. In $\bigvee \mathrm{H}-\mathrm{PSO}$, on the other hand, the individuals on the upper levels move faster. In order to investigate the consequences, the number of swaps that occur during a run have been measured. In $\bigwedge \mathrm{H}-\mathrm{PSO}$ most of the swaps take place at the top of the hierarchy. In the test runs the ratio of swaps that occur between levels 0 and 1 to swaps that occur between levels 1 and 2 is approximately 0.8 to 1, depending on the objective function. The same ratio for $\bigvee \mathrm{H}-\mathrm{PSO}$ is much lower and only approximately 0.2 to 0.4. The ratio of swaps between level 0 and 1 to swaps between 1 and 2 is displayed in Fig. 11 for $\bigwedge \mathrm{H}-\mathrm{PSO}$ and $\bigvee \mathrm{H}-\mathrm{PSO}$.

D. Diversity

The results about the diversity within the different subswarms of H-PSO are presented in this subsection. The diversity measure introduced in Section IV-B is taken for H-PSO with parameter values $m = 31$, $d = 5$ and $h = 3$. The diversity that was measured for the different test functions within the five subtrees of the H-PSO and within a randomly selected subset is displayed in Fig. 12 and Fig. 13.

For the unimodal test functions (Fig. 12), the diversity within the randomly selected subset is almost identical to the diversity within the subtrees. The diversity values are especially similar for the Rosenbrock function. Since there is only one local optimum for these functions, the subswarms of H-PSO algorithm do not concentrate on different regions of the search space. This is clearly different for the multimodal functions in which the diversity among the particles in each subtree is distinctly smaller than for particles in a random subswarm of the same size. This demonstrates that the particles of different subtrees concentrate their search effort to different areas of the search space—in multimodal environments—as intended by the design of H-PSO.

The number of swaps that occurred anywhere in the hierarchy over the entire run of 10 000 iterations has also been measured. This number is significantly larger for the unimodal test functions (>8000) and the total number of swaps up to a certain iteration increases almost linearly with the number of iterations. For the multimodal functions, the observed total number of swaps is less than 2200. Except for the Rastrigin function, the main fraction of all swaps is done very early during a run [99% of all swaps are been done until iteration 1680 (Griewank), 2470 (Schaffer), and 2240 (Ackley)]. Roughly speaking, at these iterations the behavior of the diversity curves change (see Fig. 13). For the Griewank and Ackley function, this behavior looks very similar, the diversity distance between subtrees and random subset increases, because the subtrees concentrate on different parts of the search space after an initial phase. Only very few particles move between subtrees then. For the Rastrigin and Schaffer function, the subtrees concentrate the search on different regions of the search space nearly right from the start.

VI. Conclusion

In this paper we have introduced a hierarchical version of PSO, called H-PSO, in which the particles are arranged in a dynamic hierarchy. This hierarchy gives the particles different influence on the rest of the swarm with respect to their current fitness. Hierarchies with different tree topologies have been studied for a standard set of test functions (Sphere, Rosenbrock, Rastrigin, Griewank, Schaffer's f6, Ackley), and have been compared to several variants of PSO. The H-PSO algorithm performed good on all of the considered test functions regardless of their type (unimodal or multimodal).

Moreover, a variant of H-PSO (AH-PSO) with a dynamically changing branching degree of the tree topology has been introduced which could improve the performance of H-PSO. Another extension of H-PSO is to use different values for the inertia weight w_k of the particles according to their level in the hierarchy. It has been shown that this algorithm is able to reach a specified goal for every test function (except the Rastrigin function) faster than all other variants of PSO.

In order to better understand the observed optimization behavior of H-PSO, the diversity of different subswarms of H-PSO and the number of swaps that occur within the hierarchy have been examined. These observations indicate that, in spite of the dynamic nature of the hierarchy, the subtrees are able to specialize to certain regions of the search space in case of multimodal functions.

One interesting topic for future research is to identify conditions that could trigger the decrease of the branching degree for AH-PSO so that it could autonomously adapt to the state of the optimization process.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Networks*, vol. 4, 1995, pp. 1942–1947.

[2] *Handbook of Metaheuristics*, F. Glover and G. Kochenberger, Eds., Kluwer , Norwell, MA, 2002.

[3] V. Cerný, "A thermodynamical approach to the traveling salesman problem," *J. Optim. Theory Applicat.*, vol. 45, no. 1, pp. 41–51, 1985.

[4] S. Kirkpatrick, C. D. Gelatt Jr., and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, pp. 671–680, 1983.

[5] J. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, MI: Univ. of Michigan Press, 1975.

[6] L. J. Fogel, A. J. Owens, and M. J. Walsh, *Artificial Intelligence Through Simulated Evolution*. New York: Wiley, 1966.

[7] I. Rechenberg, *Evolutionsstrategie—Optimierung technischer Systeme nach Prinzipien der biologischen Information*. Freiburg, Germany: Fromman Verlag, 1973.

[8] H.-P. Schwefel, *Numerical Optimization of Computer Models*. Chichester, U.K.: Wiley, 1981.

[9] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: optimization by a colony of cooperating agents," *IEEE Trans. Syst., Man, Cybern. B*, vol. 26, no. 1, pp. 29–41, Feb. 1996.

[10] M. Dorigo, G. Di Caro, and L. M. Gambardella, "Ant algorithms for discrete optimization," *Artif. Life*, vol. 5, no. 2, pp. 137–172, 1999.

[11] M. Dorigo and T. Stützle, *Ant Colony Optimization*. Cambridge, MA: MIT Press, 2004.

[12] M. Guntsch and M. Middendorf, "A population based approach for ACO," in *Proc. 2nd Eur. Workshop on Evolutionary Computation in Combinatorial Optimization (EvoCOP-2002)*, vol. 2279, 2002, pp. 72–81.

[13] J. Kennedy and R. C. Eberhart, "A discrete binary version of the particle swarm optimization," in *Proc. Conf. Systems, Man, and Cybernetics*, 1997, pp. 4104–4109.

[14] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. Congr. Evolutionary Computation (CEC 2002)*, 2002, pp. 1671–1676.

[15] P. N. Suganthan, "Particle swarm optimizer with neighborhood operator," in *Proc. Congr. Evolutionary Computation (CEC 1999)*, 1999, pp. 1958–1962.

[16] J. Kennedy, "Stereotyping: improving particle swarm performance with cluster analysis," in *Proc. Congr. Evolutionary Computation (CEC 2000)*, 2000, pp. 1507–1512.

[17] Y. Shi and R. C. Eberhart, "Parameter selection in particle swarm optimization," in *Proc. Evolutionary Programming VII*, vol. 1447, 1998, pp. 591–600.

[18] T. Krink, J. S. Vesterstrøm, and J. Riget, "Particle swarm optimization with spatial particle extension," in *Proc. IEEE Congr. Evolutionary Computation (CEC 2002)*, 2002, pp. 1474–1479.

[19] T. M. Blackwell, "Swarms in dynamic environments," in *Proc. Genetic and Evolutionary Computation (GECCO 2003)*, vol. 2723, 2003, pp. 1–12.

[20] J. Riget and J. S. Vesterstrøm, "A Diversity-Guided Particle Swarm Optimizer—The ARPSO," Dept. Comput. Sci., Univ. of Aarhus, Aarhus, Denmark, Tech. Rep. 2002-02, 2002.

[21] A. Silva, A. Neves, and E. Costa, "An empirical comparison of particle swarm and predator prey optimization," in *Proc. 13th Irish Int. Conf. Artificial Intelligence and Cognitive Science*, vol. 2464, 2002, pp. 103–110.

[22] J. Kennedy, "Small worlds and mega-minds: effects of neighborhood topology on particle swarm performance," in *Proc. Congr. Evolutionary Computation (CEC 1999)*, vol. 3, 1999, pp. 1931–1938.

[23] K. Veeramachaneni, T. Peram, C. Mohan, and L. A. Osadciw, "Optimization using particle swarms with near neighbor interactions," in *Proc. Genetic and Evolutionary Computation (GECCO 2003)*, vol. 2723, 2003, pp. 110–121.

[24] J. Kennedy and R. Mendes, "Neighborhood topologies in fully-informed and best-of-neighborhood particle swarms," in *Proc. IEEE Int. Workshop on Soft Computing in Industrial Applications*, 2003, pp. 45–50.

[25] J. S. Vesterstrøm, J. Riget, and T. Krink, "Division of labor in particle swarm optimization," in *Proc. Congr. Evolutionary Computation (CEC 2002)*, 2002, pp. 1570–1575.

[26] M. Løvbjerg and T. Krink, "Extending particle swarm optimizers with self-organized criticality," in *Proc. Congr. Evolutionary Computation (CEC 2002)*, 2002, pp. 1588–1593.

[27] J. Kennedy and W. M. Spears, "Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator," in *Proc. Int. Conf. Evolutionary Computation*, 1998, pp. 78–83.

[28] I. C. Trelea, "The particle swarm optimization algorithm: convergence analysis and parameter selection," *IPL: Inform. Process. Lett.*, vol. 85, 2003.

[29] F. van den Bergh, "An Analysis of Particle Swarm Optimizers," Ph.D. dissertation, Dept. Comput. Sci., Univ. Pretoria, Pretoria, South Africa, 2002.

Stefan Janson received the diploma in computer science from the University of Karlsruhe, Karlsruhe, Germany, in 2002.

He is currently working on the German Research Council (DFG) project "Methods of Swarm Intelligence on Reconfigurable Architectures" at the University of Leipzig, Leipzig, Germany.

Martin Middendorf (M'98) received the diploma degree in mathematics and the Dr.rer.nat. degree from the University of Hannover, Hannover, Germany, in 1988 and 1992, respectively. He also received the Habilitation degree in 1998 from the University of Karlsruhe, Karlsruhe, Germany.

He has previously been with the University of Dortmund, Dortmund, Germany, and the University of Hannover as a Visiting Professor of Computer Science. He was a Professor of Computer Science at the Catholic University of Eichsttt, Eichsttt, Germany. Currently he is a Professor for Parallel Computing and Complex Systems with the University of Leipzig, Leipzig, Germany. His research interests include reconfigurable architectures, parallel algorithms, algorithms from nature, and bioinformatics.