Leveraged Neighborhood Restructuring in Cultural Algorithms for Solving Real-World Numerical Optimization Problems

Mostafa Z. Ali, *Senior Member, IEEE*, Ponnuthurai N. Suganthan, *Fellow, IEEE*, Robert G. Reynolds, *Senior Member, IEEE*, and Amer F. Al-Badarneh

***Abstract*—Many researchers have developed population-based techniques to solve numerical optimization problems. Almost none of these techniques demonstrate consistent performance over a wide range of problems as these problems differ substantially in their characteristics. In the state-of-the-art cultural algorithms (CAs), problem solving is facilitated by the exchange of knowledge between a network of active knowledge sources in the belief space and networks of individuals in the population space. To enhance the performance of CAs, we restructure the social fabric interconnections to facilitate flexible communication among problem solvers in the population space. Several social network reconfiguration mechanisms and types of communications are examined. This extended CA is compared with other variants of CAs and other well-known state-of-the-art algorithms on a set of challenging real-world problems. The numerical results show that the injection of neighborhoods with flexible subnetworks enhances performance on a diverse landscape of numerical optimization problems.**

***Index Terms*—Cultural algorithm (CA), evolutionary social structures, knowledge integration, knowledge swarming, numerical optimization.**

I. Introduction

SOLVING numerical optimization problems is a challenging research endeavor in many scientific areas. Many optimization techniques and search algorithms have drawn their motivation from evolution and social behavior. These include ant colony optimization [1], genetic programming [2], genetic algorithms (GAs) [3], cultural algorithms (CAs) [4], evolutionary programming (EP) [5], differential evolution [6], particle swarm optimization (PSO) [7], simulated annealing [8], artificial neural networks [9], and artificial immune systems [10], among others.

Of these, CAs are particularly useful for problems that require extensive domain knowledge in order to produce a solution. In contrast to traditional population-based approaches, problem solving in CAs is directed by the belief space. Individuals in the population space are recruited and controlled by knowledge sources (KSs) in the belief space.

CAs are computational models derived from the cultural evolution process in nature. CAs have shown their ability when used to tackle problems from different application domains [11]–[14], [16]–[20].

In CAs [12], [13], problem solving is facilitated by the exchange of knowledge between a set of five KSs in the belief space and networks of individuals in the population space. In order to improve the effective propagation of knowledge between the individuals in the population, the extension of the CA proposed here explores the restructuring of the social fabric (social network) [11] of connections that link the individuals in the population in the population space. This restructuring is done through the use of dynamic neighborhood topology. The KSs in the belief space of the CA framework can weave a networked “fabric” of individuals that are searching the problem landscape. The population space is composed of a set of subgroups “tribes,” which represent the basic building blocks of this population of problem solvers. In anthropological terms, tribes are the basic component of a segmented society where small groups are connected to a larger network through kinship relations. In a CA, tribes are subsets of the population exploring different regions of the fitness landscape.

The tribal subgroups can be merged with each other using various regrouping schemes. The exchange of information among the individuals in the entire population starts when the initial fabric or network is woven by the KSs. These KSs represent different types of search heuristics that utilize different knowledge structures constructed from collected information during the search. Unlike current multiswarm PSOs, our tribal subgroups are dynamically forming within all layers of the established multilayered social networks. The size of each tribe is small compared to the entire population in order to enhance diversity among the individuals within the population. This small size is in keeping with the small world network characteristic of social influence groups, affinity groups, and voter groups. Here, algorithms taken from the literature which build small world networks out of a collection of subnetworks are used to combine the tribal subnetworks.

Manuscript received July 22, 2014; revised December 30, 2014, March 11, 2015, and April 28, 2015; accepted June 20, 2015. Date of publication June 25, 2015; date of current version March 29, 2016.

M. Z. Ali and A. F. Al-Badarneh are with the Department of Computer Information Systems, Jordan University of Science and Technology, Jordan 22110, Irbid (e-mail: mzali.pn@gmail.com; amerb@just.edu.jo).

P. N. Suganthan is with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798 (e-mail: epnsugan@ntu.edu.sg).

R. G. Reynolds is with the Department of Computer Science, Wayne State University, Detroit, MI 48202 USA (e-mail: robert.reynolds@cs.wayne.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2015.2450018

These small-sized formations of tribes in the population perform the search in the problem landscape using their own fitness information. However, some of the tribe formations may get trapped by converging to a local optimum due to the accelerated convergence behavior of CA [11], [14]–[15]. So, selected experiences are sporadically injected into the overall population of tribes in order to enhance its diversity in all layers of the network. Hence, several tribal regrouping schedules are used in order to produce a vigorous and varying neighborhood structure.

In this manner, the information acquired by each tribe is bartered among the other tribes to which it is connected. This interaction promotes the diversity of the whole population of agents which helps to avoid premature convergence. For the purpose of achieving enhanced performance on large-scale problems, subregional search phases steered by the knowledge operators in the belief space component are incorporated into each tribe as a further contingency measure. This process makes it possible to retain the diversity of the whole population. This allows the system to learn about certain networks that produce efficient social structures in the evolutionary belief component that can be utilized in order to produce more successful individuals in the problem landscapes.

The remainder of this paper is organized as follows. Section II describes the modified heterogeneous multilayered social fabric with network restructuring. In Section III, this restructuring process is elaborated upon and analyzes the aggregation procedure of the new influence function. The experimental framework is presented in Section IV. The results are discussed in Section V and the conclusion is drawn in Section VI.

II. Novel Multilayered Influence Function With Neighborhood Restructuring

A. Cultural Algorithm

CAs consist of a population and belief components and a set of communication protocols. The population component supports any population-based computational model such as PSO, EP, and GA. The belief space serves as a knowledge repository acquired over generations. The basic CA framework is given in Fig. 1 [4], [13] where the algorithm initializes the population space $P(t)$ and the belief space $B(t)$ at $t = 0$. The evolution then continues until a predetermined condition is satisfied. Both spaces are linked through communication protocols. The first one states the rules about individuals that contribute to $B(t)$ with their experiences (acceptance), and the other communication protocol specifies how $B(t)$ can influence new individuals (influence of each KS as detailed in Section II). Individuals are evaluated at the end of each loop via the objective function (denoted as obj in Fig. 1) and the individuals that will be used to update the belief component $B(t)$ is determined using the acceptance function. The experiences of the selected individuals will be used to update $B(t)$ using the adjust function. The selection of individuals in the successive generation is controlled using the selection function.

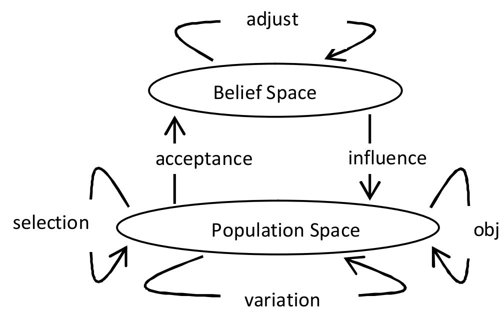

Fig. 1. Computational framework of CA.

Five types of cultural knowledge have been identified in the belief space [4], [13]: the domain KS (KS_D), the normative KS (KS_N), the situational KS (KS_S), the topographical KS (KS_T), and the history or temporal KS (KS_H). Each different KS can record different information about the evolutionary learning process, and each can influence it in its own way. Reynolds [4] and Peng [13] suggested that this set of KSs is considered complete in the sense that any aspect of cultural knowledge can be represented as a combination of a subset of this complete set. Some of these KSs can serve to explore the search space (topographic knowledge and normative knowledge) and their results attract specialists that exploit the search space (situational and domain KSs) [4], [13], [21]. Historical knowledge plays an important role when the problem is of dynamic nature [13]. The notation for optimization problems will be presented in general to match the concept of the proposed work. A global optimization problem can be summarized as follows: $\min(f(X))$, where $f(X)$ is the fitness function and $X = \langle x_1, x_2, \ldots, x_D\rangle$ is the set of design variables, such that $x_j \in [l_j,\ u_j]$, $j = 1, 2, \ldots, D$. D is the number of design variables, and l_j and u_j are the lower and upper bounds for design variable x_j, respectively. These bounds are as specified in the definition of the problem. In this paper, only static high-dimensional landscapes are the targets of optimization. Thus, there is no need for domain or history knowledge in these investigations here [13], [21], [22].

B. Belief Space

In this paper, the belief space contains the three KS components mentioned above. Each of these KSs represents the knowledge acquired during the optimization process. These KSs are described next. The representation of these KSs has been adapted for high-dimensional problems in the following fashion.

1) The situational KS interprets individual experiences using a set of stored exemplary cases. This encourages individuals to move toward superior exemplars in the population during the search process. Situational knowledge also supports local search around the exemplars. In this paper, situational knowledge stores the overall best individual and is used as an exemplar for other individuals to follow. The data structure for this KS is represented as $S_E = <x_1,\ x_2,\ \ldots,\ x_D | f(X)>$. For each exemplar, E; the structure stores the value of each parameter for this exemplar and its fitness value,

specified after the "|" in this representation. Situational knowledge is updated by replacing the exemplar by the population's best individual ($X_{\text{best}}(t)$) if it outperforms the current exemplar at generation t. The update equation is

$$E(t+1) = \begin{cases} X_{\text{best}}(t), & f(X_{\text{best}}(t)) > f(E(t)) \\ E(t), & \text{otherwise.} \end{cases} \quad (1)$$

2) Normative knowledge is defined as the promising parameter ranges that provide guidelines and standards for individuals' behaviors. These norms are adjusted based on individual's performances during the search. The structure of this exploratory knowledge contains an interval for each parameter. The normative data structure consists of a lower bound lb_j and an upper bound ub_j for each variable. $\text{PU}_j(t)$ and $\text{PL}_j(t)$ are the values of the fitness function associated with these bounds at time t. The update process for the lower bound of the jth variable at time $t+1$ (modifying the bounds based on performance) is

$$\text{lb}_j(t+1) = \begin{cases} x_{i,j}(t), & x_{i,j}(t) \le \text{lb}_j(t) \text{ or } f(X_i) < \text{PL}_j(t) \\ \text{lb}_j(t), & \text{otherwise} \end{cases} \quad (2)$$

$$\text{PL}_j(t) = \begin{cases} f(X_i), & x_{i,j}(t) \le \text{lb}_j(t) \text{ or } f(X_i) < \text{PL}_j(t) \\ \text{PL}_j(t), & \text{otherwise} \end{cases} \quad (3)$$

where the ith individual can affect the lower bound for variable j at time t. The update process for the upper bound of the interval is done in a similar manner, as

$$\text{ub}_j(t+1) = \begin{cases} x_{i,j}(t), & x_{i,j}(t) \ge \text{ub}_j(t) \text{ or } f(X_i) < \text{UL}_j(t) \\ \text{ub}_j(t), & \text{otherwise} \end{cases} \quad (4)$$

$$\text{PU}_j(t) = \begin{cases} f(X_i), & x_{i,j}(t) \ge \text{ub}_j(t) \text{ or } f(X_i) < \text{PU}_j(t) \\ \text{PU}_j(t), & \text{otherwise.} \end{cases} \quad (5)$$

3) The topographic knowledge is an exploratory KS which divides the whole landscape into cells and keeps track of the best individual in each cell.

The topographic KS is used to divide the current search regions into subregions along each dimension of the problem. Each region will be divided into two new subregions in each generation of the optimization process. This limitation on the number of children for each cell is for the sake of a more efficient memory management in the presence of high dimensionality. A *k-d* tree representation for this knowledge structure is available for higher dimensions as well [1]. In the case of high dimensionality, the *k-d* tree is used to subdivide a promising node in a similar fashion for any of the k dimensions.

The topographic structure of the original CA organizes the cells of the search space into a hierarchical tree structure, where the search space is initially divided into L cells. This is formulated as

$$\text{CS}(t) = \langle C_1(t), C_2(t), \ldots, C_L(t) \rangle \quad (6)$$

where $\text{CS}(t)$ is the topographic knowledge structure and is composed of regions $C_r(t)$. The description of a cell in the feasible distribution of branched cells is given as

$$C_r(t) = \left\langle L_r(t), U_r(t), \text{state}_r(t), d_r(t), \text{pt}_r(t) \right\rangle \quad (7)$$

where the lower and upper bounds of a certain cell are represented as

$$L_r(t) = \langle l_1(t), l_2(t), \ldots, l_D(t) \rangle$$

and

$$U_r(t) = \langle u_1(t), u_2(t), \ldots, u_D(t) \rangle.$$

Each cell in this data structure contains a lower bound and an upper bound for each of the D variables, indicating the ranges associated with the best solutions found in that cell so far. $\text{state}_r(t)$ is the state of a given cell r at time t, and is represented as

$$\text{state}_r(t) = \begin{cases} H, & f\left(\hat{X}_r(t)\right) > \bar{f}\left(\hat{X}(t)\right) \\ \text{NE}, & \forall X \in P(t) : X \notin C_r(t) \\ L, & f\left(\hat{X}_r(t)\right) \le \bar{f}\left(\hat{X}(t)\right) \end{cases} \quad (8)$$

where H is the most promising cell for further exploration for enhancing the solution. $f(\hat{X}_r(t))$ is the fitness of the best individual in cell r, and $\bar{f}(\hat{X}(t))$ is the average of all the fittest individuals in all cells of the search space. NE is a cell that has not been explored. This means, without loss of generality, that there is no individual in cell $C_r(t)$, at time t.

Cells are ordered according to how promising they are when explored, and those with the highest rank are marked as H. Moreover, each cell has a depth that is denoted as $d_r(t)$ and represents the number of times that a search region is subdivided for further exploration. Memory usage increases during the search when cells are partitioned further. Hence, in a manner that is different from the original CA, we will use a branching restriction of $d_r(t) \le \bar{d}$ for this paper. Each nonleaf cell uses a pointer to its children, denoted as $pt_r(t)$.

The update process is given as

$$\langle C_1(t+1), \ldots, C_L(t+1) \rangle = \begin{cases} \langle C_1(t), \ldots, C_L(t) \rangle \cup \left\langle C_{r\_1}(t), C_{r\_2}(t), C_{r\_3}(t), C_{r\_4}(t) \right\rangle, \\ \qquad \text{if } f\left(X_{i,r}(t)\right) > f\left(\hat{X}_r\right) \\ \langle C_1(t), \ldots, C_L(t) \rangle, \quad \text{otherwise} \end{cases} \quad (9)$$

where $\langle C_1(t+1), \ldots, C_L(t+1) \rangle$ is the set of cells in the feasible distribution at generation $t+1$, $r \in \text{branch}(t)$. $f(X_{i,r}(t))$ is the fitness of individual i in cell r, at time t. The set $\langle C_{r\_1}(t), C_{r\_2}(t), C_{r\_3}(t), C_{r\_4}(t) \rangle$ represents the subcells that result from the branching process of cell $C_r(t)$. The branching process can take place in more than one cell simultaneously.

C. Novel Two-Class Architecture of the Tribal CA

The influence function in CAs is an important factor that affects how well individuals can find promising regions. Here, the influence function in CA utilizes a layer-based social fabric of connections to form groups. This influence function uses heterogeneous and adaptive restructuring for the groups. This serves to assign the most suitable formations of subgraphs or layers of connections to the optimization of different problems. The details of this influence function are given next.

The influence function developed in this paper, named tribal sociocultural algorithm with neighborhood restructuring (T-SCANeR), was motivated by Reynolds *et al.* [19] in Archeology. The technique uses four imperative concepts: 1) layer; 2) class; 3) stage; and 4) structure. In this context, there are several network layers, where each corresponds to a defined relation R between the agents. The population is arranged into two classes and the optimization is broken down into three epochs (stages). The system restructures the formations of all tribes in the population whenever the optimization process is marked as "stagnated."

In the multilayered sociocultural system, the search process is initialized with a population that is divided into z tribes of agents. Every tribe contains h agents. Each tribe will have two best exemplars. The first is the best of all followers of the explorer KSs in the tribe ($t\text{Elite}_{\text{expr}}$), and the second is the best of all followers of the exploiter KSs in the tribe ($t\text{Elite}_{\text{expt}}$). Each tribe will have a certain topology that determines its shape in the social fabric structure at any time during the search. The hierarchal structure in this design will contain the set of all agents in tribes as the rudimentary class, whereas the set of all tElite-pairs of all tribes will form the advanced class.

D. Three Stages of Problem Solving of the Tribal CA

The influence function passes through three stages, which comprise its problem-solving process. These stages are the seclusion stage, the rapport stage, and the cohesive stage. Each of these stages consumes a portion of the available function evaluations (FEs). During the seclusion stage, the algorithm guarantees that groups will have sufficient development time (Wsize) to avoid the possibility of premature convergence. In this stage, each of the z tribes will function as an independent basic CA model and no information is exchanged between any pair of tribes.

Next, in the rapport stage it employs the two-class structure, illustrated in Fig. 2. In this stage, the h tribe members form the rudimentary class, and the best individuals from each group ($t\text{Elite}_{\text{expr}}$ and $t\text{Elite}_{\text{expt}}$) constitute the advanced class. In this epoch, the agents will exchange information with agents in the tribe based upon the current topology. Generally, the two best exemplars (explorer and exploiter streams) from each tribe will have a higher impact on the members of that tribe than others in the tribe.

An EP module is used as a population space for this paper. The selection of the appropriate parents in the mutation process is determined by the KSs that are used in the belief space.

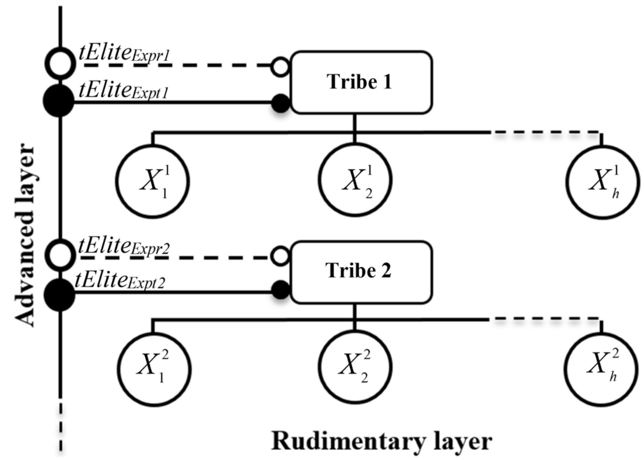

Fig. 2. Two-class taxonomy of the tribal CA.

The notation $X_i^k(t)$ will be used to refer to the ith individual of the kth group at time t, to represent the update process. The update process for the situational KS is given as

$$E(t+1) = \begin{cases} X_{t\text{Elite}}^k(t), & f\left(X_{t\text{Elite}}^k(t)\right) > f(E(t)) \\ E(t), & \text{otherwise} \end{cases} \quad (10)$$

where $x_{t\text{Elite}}^k(t)$ is $t\text{Elite}_{\text{expr}}$ of the explorer subgroup in tribe $k (k = 1, 2, \ldots, z)$, and is $t\text{Elite}_{\text{expt}}$ of exploiter subgroup in tribe k.

Normative knowledge employs a subset of individuals within its range to perform a global search and exploration of the problem landscape. An offspring individual is generated from a parent only if that parent is found to be in a feasible region of the jth dimension of this KS. In this case, the child is generated with a magnitude that is dependent upon the current feasible solution of the parent and is modified stochastically based on the parent's fitness. Otherwise, the child individual is generated at random in that feasible region based on a uniform distribution. If λ is assumed to represent $\text{ub}_j(t) - \text{lb}_j(t)$, then the mutation operator for the normative knowledge can be formulated as

$$x_{ij}'^k(t) = \begin{cases} x_{t\text{Elite},j}^k(t) + f\left(X_i^k(t)\right) \\ \quad \times \dfrac{\lambda}{\sum_{i=1}^N f\left(X_i^k(t)\right)}, & x_{ij}^k(t) \in \text{FR}_j \\ \text{lb}_j(t) + \alpha_i \cdot (\lambda), & x_{ij}(t) \notin \text{FR}_j \end{cases} \quad (11)$$

where $X_i^k(t)$ is the ith parent before the mutation takes place. $x_{t\text{Elite},j}^k(t)$ is the value of the jth parameter of tElite individual in tribe k. $X_i'(t)$ is the ith child produced as a result of the mutation. $\text{FR}_j = [\text{lb}_j^t, \text{ub}_j^t]$, records the bounds of the jth variable. α_i is a uniform random variable, $\alpha \sim U(0,\ 1)$.

In a similar manner, the topographic KS will generate an offspring individual from a parent individual produced by the mutation process, if the parent is found in a cell marked with an H. Otherwise, if that cell is marked with NE or L, then new parent individuals are selected at random from higher-ranked cells, and are used to generate the new children. The mutation

TABLE I
POPULATION-BASED OPTIMIZERS WITH UPDATE-RULE HETEROGENEITY

Characteristics	Heterogeneity Type
Small world	Static [25]
Topologies: different average degree	Static [26]
Tree-like directed topologies	Dynamic [27]
Arbitrary neighborhood size	Dynamic [28]
Distance-based neighborhoods (arbitrary)	Adaptive [29]
Interparticle distance-based neighborhoods	Adaptive [30]
Fully informed PSO	Static [31]
Comprehensive learning PSO	Adaptive [32]
Dynamic multi-swarm PSO	Dynamic [33]

TABLE II
POPULATION-BASED OPTIMIZERS WITH ASSORTED NEIGHBORHOODS

Characteristics	Heterogeneity Type
Charged and Neutral particles	Static [34]
Quantum and Neutral particles	Static [34]
Cooperative and defective particles	Adaptive [35]
Predator/Prey particle	Static [36]
Fitness-distance ratio	Static [37]

operator is given as

$$x'^{k}_{ij}(t) = \begin{cases} x^{k}_{t\mathrm{Elite},j}(t) + \alpha_i(u_r(t) - l_r(t))\Big/2, \\ \qquad \left(x^{k}_{ij}(t) \notin C_r(t)\right) \wedge (\mathrm{state}_r(t) \neq L) \\ x^{k}_{t\mathrm{Elite},j}(t) + \alpha_i\sqrt{f\left(X^{k}_{i}(t)\right)}\Big/D, \\ \qquad \left(x^{k}_{ij}(t) \in C_r(t)\right) \wedge (\mathrm{state}_r(t) = H). \end{cases} \tag{12}$$

Diversity in this stage is well preserved, because the connections are still limited to intratribal ones. Each of the z tribes in this stage work as an independent CA model with its own social fabric influences functions that serve to distribute knowledge among the tribe members.

In the last stage, the cohesive stage, tribes will exchange information about the best experiences during the search process in that generation when all tribes are recombined into one CA model.

The purpose of the threefold process is to preserve diversity and maintain a balance between exploration and exploitation by allowing the population to follow the "best of both worlds." This can preserve the merits of either stream ($t\mathrm{Elite}_{\mathrm{Expr}}$ and $t\mathrm{Elite}_{\mathrm{Expt}}$ for each tribe), even if their performance is lower than that for another tribe during a certain search stage.

The topology that connects the agents in each tribe can be modified and reformed every G generations when a tribe is flagged as trapped at a local optimum. This causes the neighborhood of each tribe to vary based on the performance of the set of all agents in this tribe in order to avoid stagnation. This works as a diversity preserving-measure. The search process is marked as stagnated when the new best solution does not change from the previous best solutions. Here, this occurs when the difference between the tribe's newly obtained solution and that of the previous best is smaller than $\varepsilon = 10^{-8}$. In that case, it is marked as being in a stagnation state.

III. NEIGHBORHOOD RESTRUCTURING IN THE SYSTEM

In the basic version of this paper, knowledge operators are not aware of the constructed network at the population-level. Variation among the distinct features of agents' social structure can evolve into a taxonomy based on some kind of heterogeneity [20], [23], [24]. A summary of other known evolutionary algorithms with update rule heterogeneity with respect to tribes is shown in Table I. In the basic version of this paper, different individuals are guided by different KSs and hence update rules, which affect the way these individuals perform their search. Update rule heterogeneity, whether static, dynamic, or adaptive as shown in Table I, makes it possible to have groups of specialized individuals with different (but complementary) roles.

Normally, topologies are constructed as either entirely regular or entirely random. Moreover, various biological structures and real-world networks that are social in nature are classified at a complexity level that lies between the two extremes. Table II presents a summary of evolutionary optimizers with assorted neighborhoods.

This paper investigated some models and frameworks of networks that are either regular, random, or can be tuned to lie somewhere in the middle. This modification produces regular networks that are "rewired" in order to escalate the degree of disorder or chaos in the system. Here, a strategic rewiring approach is introduced in Section III-A. It will be used along approaches such as layered-Delaunay triangulation (DT) [38], and the arbitrary renovating approach [38].

A. Strategic Restructuring

Tactical restructuring is a new approach that produces constructed neighborhood structures that exhibit an abrupt type of dynamism. Undirected connections are presumed in this paper. The topologies of all tribes in the population will maintain a specific topological structure until the agents in the population get trapped at a local optimum for a number of generations. In this case, tribal topologies will then be reinitialized in order to create a pulse for a new search using a new mix of information.

Here, topologies are chosen based on the performance of agents during the search. The proper topology for each scenario will support the objective of the search mode, namely exploration or exploitation. Different types of topologies with varying number of connections are supported in the system.

1) Ring topology (lbest) has the fewest number of connections where each individual is connected to exactly two neighbors [Fig. 3(a)].
2) Von Neumann topology forms a 2-D grid with neighbors of each individual to the north, south, east, and west [Fig. 3(b)].
3) Hybrid tree topology in which a root node is connected to one or more other nodes that are one level lower in the hierarchy [Fig. 3(c)].

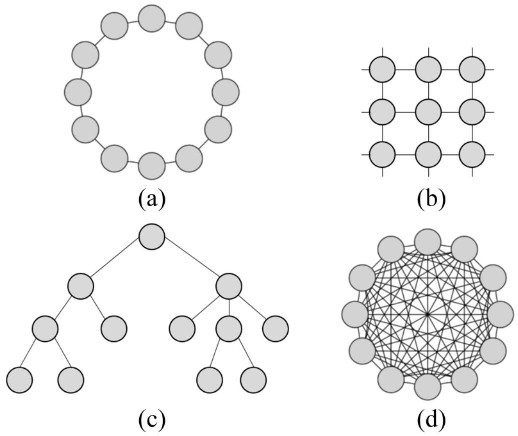

Fig. 3. Available topologies in the T-SCANeR model. (a) Ring topology (lbest). (b) Flattened representation of the Von Neumann neighborhood. (c) Hybrid tree topology. (d) Global topology (gbest).

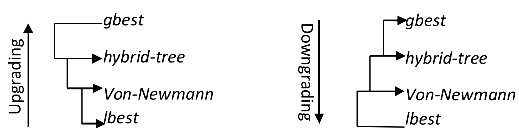

Fig. 4. Upgrading/downgrading scale for gradually enhancing the tribe's social structure based on tightness of interaction.

4) Global topology (gbest) has the highest number of connections where each individual is connected to every other individual in the group [Fig. 3(d)].

If the agents of a tribe produce stagnation at a local optimum, then the best choice would be a topology with fewer connections such as the ring topology for exploration search mode. The fabric of the entire population will then be changed through restructuring the topology of the stagnated tribe, and hence increase the exploration radius of agents. This helps in reducing the deadlock in connections, and produces movement. When agents need more information, they can move into a mesh topology, and ultimately to a global topology, which supports the exploitation search mode. This process works based on our proposed upgrading/downgrading scale. Whenever the solution does not improve for M_{thresh} generations, then the algorithm switches the topology to some other topology with fewer connections as illustrated in Fig. 4, to enhance the social information exchange. Fig. 4 shows the direction of upgrading/downgrading among the available topologies that can be used to increase/decrease connections. The number of connections can be gradually increased (left part of Fig. 4) by moving in the direction of the arrow in the corresponding figure, as in lbest → Von-Newmann → hybrid-tree → gbest, as needed. The same is true about gradually decreasing the number of connections (right part of Fig. 4) as in: gbest → hybrid-tree → Von-Newmann → lbest.

As will be explained in Section III-B, the interaction and overlapping of the search regions of the KSs play an important part in determining the appropriate restructuring of connections.

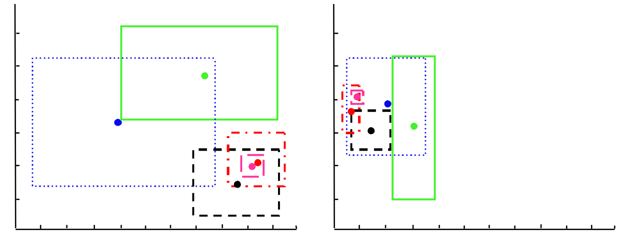

Fig. 5. Patches (bounding boxes) associated with the different KSs. Explorers are taking the lead in the figure to the left, while exploiters in the right figure start to fine-tune the boundary, with a tighter search, in which they produce their follower individuals.

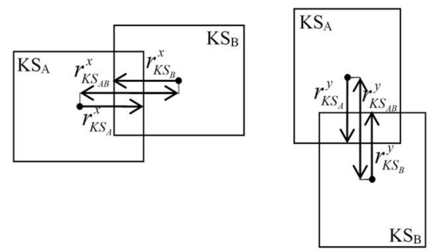

Fig. 6. Breaking down the interaction between the KSs patches into key components.

B. *Effect of Restructuring on the Synergy of Knowledge Sources*

The bounding box for a given KS contains a certain percentage of those individuals generated or controlled by that KS. The positioning of these patches (bounding boxes), with a dynamically changing central tendency and standard deviation, serves to generate individuals in the associated zone. The slow progress during the search is due to tighter patches of individuals under control of each KS, depending on the search progress, as can be seen in Fig. 5.

As stated above, the restructuring of tribes in the population is triggered by the best solutions produced by the elites of both streams of problem solvers, the elite of explorers ($t\text{Elite}_{\text{Expr}}$) and elite of exploiters ($t\text{Elite}_{\text{Expt}}$), for a particular tribe. The function, tribeTriggerCriteria(tribe), in the pseudocode of Table III checks to see if the number of generations S_k over which tribe S_k's best individuals have not changed exceeds a certain threshold (M_{thresh}). If so, then the tribe will switch to a topology that reflects the social factor (S_{Fact}) based on Fig. 4. The social factor describes the new neighborhood over which agents are allowed to exchange information. This depends on the search mode, as will be explained next.

The neighborhood restructuring procedure for the algorithm is directly linked to S_k, and the average distance between centers of the KSs patches. Whenever the overlap between the boxes of the KSs is large, and S_k is larger than M_{thresh}, then the system gradually downgrades (in terms of number of connections) the topology of the selected stagnated tribes.

This is shown in step 1.2.1 in Table III. Otherwise, it upgrades (according to the scale shown in Fig. 4) to find the solution. The overlap of the patches plays an important role here, since it determines where new individuals are to be

TABLE III
T-SCANER HIGH-LEVEL OVERVIEW

Algorithm: T-SCANER

Input:
- Optimization problem – Dimensionality of the problem and the fitness values (optimal) should be provided
- Termination condition

Output: The best found solution and the position of the best individual

Step 0) Initialization: Set $t = 0$. Randomly initialize the population component $P(t)$ and the belief space $B(t)$
- **Step 0.1)** Initialize the population of N candidate solutions as described in section II
- **Step 0.2)** Initialize the knowledge sources with random followers (roulette wheel)

Step 1) for each layer α_i defined over a relation R do // We start with two relations: knowledge and topologies
- **Step 1.1)** Subdivide the population into z tribes randomly
- **Step 1.2)** For each tribe in this network in layer α_i do
  - o **Step 1.2.1) Restruct:** Use the set of all tribes as input and perform the topology modification and restructuring procedure – as discussed in subsection III.A
    - **Step R1) trigger** ← tribeTriggerCriteria(tribe)
    - **Step R2)** if $tElite_{Expr}$ == $Tribe\_BestF$ && $tElite_{Expt}$ == $Tribe\_BestF$ **then**
      - ▪ **Step R2.1)** If # of generations tribe S_k has been stagnated for is M_{thresh}
        //utilize the info in subsection III.C to determine the
        //overlapping to determine the most suitable topology
        - o **Step R2.1.1)** restructure the topology for this tribe to the topology equivalent to S_{fact} , $S_{fact} \leftarrow$ getS$_{fact}$();
        - o **Step R2.1.2)** ModifyTopologyStr(S_{fact}); // based on Fig. 4
        - o **Step R2.1.3)** $n_{S_k} \leftarrow 0$ //stagnation counter

      otherwise, if the condition in step 2.1 is not true:
      $$n_{S_k} \leftarrow n_{S_k} + 1$$
    otherwise, if the condition in step2 is not true, then: $n_{S_k} \leftarrow 0$
  - o **Step 1.2.2)** variation, evaluate and select: For each individual of the tribe update its info, and calculate fitness solutions using equations (8)-(10), and replacing $X^k_{tElite}(t)$ by $X_{Best}(t)$ (this is the best of the entire population) in Eqn(8), and replacing $x^k_{tElite}(t)$ by $x_{ij}(t)$ and $x_{rj}(t)$ in Eqns(9) and (10), respectively.
  - o **Step 1.2.3) DetermineTribeBest:** Determine the best individual of a tribe
- **Step 1.3) FindBestTribe:** The best tribe in the current network
- **Step 1.4) Secluded Stage:** Tribes will be evaluated, each as a separate basic CA model. Communication is governed by the social index $Grade_i(t)$ as explained in III.B
- **Step 1.5) Rapport Stage:** Tribes_Communication() using eqns. (10)-(12).
- **Step 1.6) Cohesive Stage:** Communication between tribes after uniting tribes into one CA model

Step 2) Aggregate layers
- **Step 2.1) Reductase:** Social fabric aggregation over the R relations $\{R_1, R_2 ... R_R\}$. Determine Liaisons of each relation R
- **Step 2.2)** Calculate the measure of each individual (represented as a node in the sociomatrix):
  $$\tau_j = \sum_{i \in Nbr(j)} m_{ij} + \psi_j \text{, as discussed in section III}$$
- **Step 2.3)** Find the sum of the individuals in the tribe based on:
  $$\tau_{S_j} = \sum_{k \in S_j} \left(\sum_{i \in Nbr(k)} m_{ik} + \psi_k \right) \text{ //measure of tribe } S_j$$
- **Step 2.4)** If S_i and S_j are two subgroups, $k \in S_i \cap S_j$; $j \in S_j \setminus S_i$; $i \in S_i \setminus S_j$, the arc weight between the two subgroups is given as:
  $$\tau_{S_i,S_j} = \sum_{a,k \in S_i \setminus S_j} m_{ak} + \sum_{\substack{k \in S_i \setminus S_j \\ b \in S_j \setminus S_i}} m_{kb} + \psi_k + \psi_b \text{ //indirect knowledge sharing between both groups}$$
- **Step 2.5)** influence: Use steps 2.1–2.4 to find the impact (affecting knowledge source) on each individual a_i of each tribe S_j, where $a_i \in S_j$ using vector voting (majority of votes wins), as discussed in subsection III.C. In case of ties, tie breaking rules can be used (KS$_{direct}$, KS$_{LFU}$, KS$_{MFU}$, KS$_{Last\text{-}used}$, KS$_{random}$)

Step 3) Terminate and produce output

generated. Individuals that are closer together have a greater chance to share knowledge and move faster to new peaks in that area.

By looking at Fig. 6, some logical relations about the interaction between the two bounding boxes associated with two KSs: KS$_A$ and KS$_B$, can be stated. For example, in 2-D assume that the radius of KS_A in the x-direction is $r^x_{KS_A}$ and the radius of KS_B in the x-direction is $r^x_{KS_B}$.

In the case that the distance from the center of the bounding box associated with KS_A to the center of the bounding box associated with KS_B ($r^x_{KS_{AB}}$) has an upper limit equal to the sum of $r^x_{KS_A}$ and $r^x_{KS_B}$, then the regions associated with these two KSs are aligned in a way that makes them share common coordinates in the x-plane, where

$$r^x_{KS_{AB}} \leq r^x_{KS_A} + r^x_{KS_B}. \tag{13}$$

The same is true in the y-plane

$$r^y_{KS_{AB}} \leq r^y_{KS_A} + r^y_{KS_B}. \tag{14}$$

In order to calculate the distance between the two centers of the bounding boxes in the x-direction, one can add the radius of the bounding box in the x-direction to the top-left x-magnitude

$$r^x_{KS_{AB}} = \left| \left(\mathrm{TL}\left(KS^x_A\right) + r^x_{KS_A}\right) - \left(\mathrm{TL}\left(KS^x_B\right) + r^x_{KS_B}\right) \right| \tag{15}$$

such that TL(.) refers to the top-left x-magnitude. The radius $r^x_{KS_A}$ is calculated by finding half the length between the top-left x-magnitude and the bottom-right x-magnitude [BR(.)]. The same is true for the y-direction. If both conditions (13) and (14) are satisfied, then the bounding boxes of the corresponding KSs are said to overlap.

The search for differences between the agents in the population is an important factor in the current search mode. This search mode depends on the fitness value of each individual agent. The social index, as proposed in [39], will be used to accomplish such a task and highlight performance differences. This social factor is given as

$$\mathrm{Grade}_i(t) = \frac{f_{\mathrm{worst}}(t) - f\left(X^k_i(t)\right)}{f_{\mathrm{worst}}(t) - f_{\mathrm{best}}(t)} \tag{16}$$

where the best and worst fitness values of both explorers and exploiters groups, at time t, are denoted as $f_{\mathrm{best}}(t)$ and $f_{\mathrm{worst}}(t)$, respectively. $f(X^k_i(t))$ is the fitness of the individual i at time t. The value of $\mathrm{Grade}_i(t)$ is used as a representation of the performance of a certain individual at time t, based on the computed fitness of the present location for that individual. $\mathrm{Grade}_i(t)$ is used to govern the communication within each tribe. Accordingly, if the calculated fitness value for agent X^k_i is ranked higher than that for agent X^k_j, then the likelihood of convergence of the X_i-neighborhood into a global optimum, in the search region, is higher than that of X^k_j. Consequently, this indicates that agent X^k_i should be renegotiating its neighborhood more frequently in order to converge to move more toward its neighbors if it is in exploitation mode, or explore broader regions if it is in exploration mode.

When $f_{\mathrm{best}}(t)$ is equal to $f_{\mathrm{worst}}(t)$, then the population of problem solvers converges onto one point during the process of searching for the optimum. In this case, we arbitrarily set the value of the social index to 1.

C. Novel Perspective of Aggregation in the Social Network

This discussion will describe the aggregation process for the influence function that determines how the appropriate KS is selected for individuals in a tribe. The description will correspond to the pseudocode given in Table III.

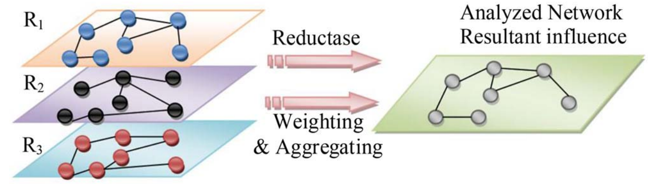

Fig. 7. Multilayer social network aggregation framework.

A Tribal network is represented as a directed weighted graph that reflects the communication $l_{n,m}$ (social link) between two agents, X^k_i and X^k_j, where

$A_i = \{X^k_i(t)\}$: set of agents, where $X^k_i(t)$ is the ith agent in tribe k at time t.

$L^k = l^k_{n,m}$: set of all links between agents in tribe k, $\forall n, m \in A_i$.

The mathematical representation of the social links is called the sociomatrix (Ω). The sociomatrix is a matrix that is indexed by the two communicating agents to represent their dimensions and is equivalent to the adjacency graph when it captures dichotomous, symmetric relationships [40]. For a given relation (R) and a set of u agents in $N = \{X_1, X_2, \ldots X_u\}$, the magnitude $\upsilon_{i,j} \in \Omega$ is equal to the value of the bond from X_i to X_j on the given relation R.

If we assume that we have q relations $R_1, R_2, \ldots, R_Q$, for the same set of agents, then the value $m^q_{ij} \in X^q$ is the value of the link from X_i to X_j on relation R^q. Fig. 7 illustrates the multilayer social network aggregation framework. In this paper, we have the special case of two relations. The first one includes connections through the KSs, and the second one includes social connections throughout formed topologies.

Groups of individuals in social networks can be categorized into two structures.

1) Dissimilar tribes linked merely via cut-outs (independent).
2) Tribes linked by liaison agents who are members in more than one relational network (multiple memberships). This will be the adopted type in [38] and [40].

Referring to the pseudocode in Table III, assume that m_{ij} is the measure, representing the knowledge on edge $l^k_{i,j}$, in tribe k, that affects individual X^k_j by X^k_i.

Moreover, τ_j is the measure on node X^k_j representing its net affecting knowledge from the belief space and other neighbors ($Nbr(j)$). ψ_j is the KS controlling node X^k_j as in the basic CA. In the rest of the discussion, the summation symbol is used to represent majority voting on the set of all KSs on the arcs connecting an individual to its neighbors. Majority voting is used to find the controlling KS, along with tie breaking rules that break ties in the vote, (TieR). Some example tie-breaking rules are: the direct influencing KS, KS_{direct}; the least frequently used KS. KS_{LFU}; the most frequently used KS, KS_{MFU}; and the KS that controlled the individual in the last iteration, $KS_{\text{Last-used}}$, or a random choice, KS_{random} [11]. The degree measure for each agent (vertex) is defined as a sum of all the weights on the arcs out of this vertex as

$$\tau_j = \sum_{i \in Nbr(j)} m_{ij} + \psi_j. \tag{17}$$

Each aggregated node will be assigned, during the aggregation step, the total of the weights of all vertices that are aggregated into a vertex. A unity weight will be assigned to an existing arc for which there is no information available.

Assume that S_j is a tribe in our population, then the measure of S_j (denoted as τ_{S_j}) is defined to be the summation of the measures of all the agents in the tribe, based on (17), as

$$\tau_{S_j} = \sum_{k \in S_j} \tau_k$$

$$\tau_{S_j} = \sum_{k \in S_j} \left(\sum_{i \in Nbr(k)} m_{ik} + \psi_k \right). \quad (18)$$

In the aggregation step, the weights of the edges between two aggregated individuals are added. If we assume that S_i and S_j are two tribes then, $a, k \in S_i \backslash S_j$; $b \in S_j \backslash S_i$.

The strength of the connection between S_i and S_j to be the sum of arcs that connect them as

$$\tau_{S_i,S_j} = \sum_{a,k \in S_i \backslash S_j} (m_{ak} + \psi_k) + \sum_{\substack{k \in S_i \backslash S_j \\ b \in S_j \backslash S_i}} (m_{kb} + \psi_b). \quad (19)$$

Consequently, for any two aggregated nodes that exist in two different groups, the measure can be defined to be a summation of two different types of edges or arcs: 1) the arcs that connect agents simply within S_i to the liaison agents and 2) arcs that link liaisons to the agents within subset S_j.

Assuming $k \in S_i$, $v \in S_j$, each subgroup, S_i, which has been marked as aggregated, is assigned a node measure that is equal to the sum of edges out of the subset, as

$$\tau_{S_i} = \sum_{v \in S_j} m_{k,v}. \quad (20)$$

The overall pseudocode of T-SCANeR is presented in Table III.

IV. Experimental Setup

A. Benchmark Functions

Figures and tables from the supplementary file have separate numbering where the name of each table starts with an "S." followed by the number. For example, Table I in the supplementary file is called Table S.I.

The problem set consists of 22 real-world problems, referred to as T01–T13, and is summarized in Tables S.I–S.V (see supplementary file). These problems were used for the IEEE-CEC2011 Competition on Testing Evolutionary Algorithms on Real-world Numerical Optimization Problems [41]. The competition's framework is used to help set the parameters for the experiments in order to allow a fair comparison with the other optimization algorithms. More details on these benchmarks are given in [41].

In these applications, finding the best solution is a challenging task when features such as high dimensionality, ill-conditioning and multimodality exist. Such features can increase the problem's complexity and make it harder to find quality solutions. The dimensionality of the problems used here ranges from 1 to 240. Several problems are extremely multimodal in nature with copious local optima, making the objective of locating the global solution a challenge to any optimization algorithm.

B. Parameter Settings

The algorithm was coded in JAVA and executed under a 2.4 GHz Intel Core i7-2720QM, 8 GB RAM, and Windows 7.

In order to create similar restrictions as those specified in the competition with regard to general parameters, the population size was set to $N = 90$, and the FEs limits used was 1.5×10^5. (Comparisons at 5×10^4, 1×10^5 FEs limits are provided in the supplementary file in Tables S.VI–S.IX.)

The CA starts with nine tribes, each with ten agents. Other parameters used in T-SCANeR include the number of feasible elites in each tribe ($n_{t\text{Elite}}$) which was limited to 2, one from each stream of explorers and exploiters for each tribe in this paper. The tie-breaking-rule (TieR), the window-size for applying the social influence with restructuring (Wsize) was set at $W\text{size} = 20$ to weave the social fabric. The proportion of the fittest agents whose experiences will be used to affect the belief space structure ($P_a \in [0, 1]$) was fixed in the experiments at 0.25 [11], [13], [17]–[19]; α_i is randomly chosen from a uniform distribution for every individual at every iteration. Triggering neighborhood restructuring was done using M_{thresh}. The social factor (S_f) is an integer value variable that takes a value between 0 and 3. Note that these numbers correspond to topologies, lbest, von-Neumann, n-star-bus, and gbest [17]. We performed 25 independent runs for each benchmark problem.

C. Initial Population and Initialization Method

In order to facilitate a fair comparison, the population of every algorithm included in the comparison was initialized using the same random seeds. Following the instructions of the competition [41], a uniform random distribution was used to initialize the populations.

V. Experimental Results and Analysis

A description of the problems and extra results are presented in a supplementary file and can be downloaded from http://www.ntu.edu.sg/home/epnsugan.

A. Sensitivity Analysis

In this section, a sensitivity study with an assessment of the algorithm's performance with respect to certain parameters is presented in Table IV and Fig. 8. The parameters to which the CAs performance is sensitive are: the threshold trigger for restructuring (M_{thresh}); the number of feasible elites of each tribe ($n_{t\text{Elite}}$) as candidates for members affecting the belief space in the following generation; the tie breaking rule used during communications (TieR); and the window size after which the social fabric is applied throughout the run (Wsize). The latter parameter is considered the rate at which information is exchanged. Note that P_a is a portion of the population space and need not be adjusted since it was taken as described in the original CA [13].

TABLE IV
SUMMARY OF AVERAGE AND THE STANDARD DEVIATION FOR 35 INDEPENDENT RUNS TESTED ON SELECTED FUNCTIONS (RANDOMLY FROM DIFFERENT CATEGORIES) T_4, $T_{11.2}$, $T_{11.4}$, $T_{11.7}$, AND $T_{11.10}$

	Mean results (standard deviation)				
M_{thresh} / Prob.	10	30	50	70	80
T4	1.5672E+01 (9.26E-01)	1.4652E+01 (4.56E-01)	1.3537E+01 (3.00E-01)	1.4821E+01 (2.96E-01)	1.5249E+01 (8.97E-01)
T11.2	1.1316E+06 (5.99E+03)	1.1296E+06 (6.38E+03)	1.0261E+06 (4.11E+03)	1.0271E+06 (5.27E+03)	1.1341E+06 (8.51E+03)
T11.4	1.8087E+04 (5.62E+01)	1.7854E+04 (3.96E+01)	1.7482E+04 (2.78E+01)	1.7631E+04 (3.77E+01)	1.7916E+04 (4.48E+01)
T11.7	1.7212E+06 (1.51E+04)	1.7106E+06 (6.39E+03)	1.6832E+06 (4.23E+03)	1.6947E+06 (5.89E+03)	1.7742E+06 (1.40E+04)
T11.10	9.2689E+05 (4.38E+02)	9.2568E+05 (5.73E+02)	9.2309E+05 (4.27E+02)	9.2578E+05 (4.12E+02)	9.2756E+06 (5.89E+02)
TieR	KS_{LFU}	KS_{Direct}	KS_{MFU}	KS_{Random}	$KS_{Last\text{-}used}$
T4	1.4068E+01 (1.76E-01)	1.3537E+01 (3.00E-01)	1.4180E+01 (5.48E-01)	1.4265E+01 (6.13E-01)	1.5246E+01 (6.84E-01)
T11.2	1.0261E+06 (4.11E+03)	1.0266E+06 (4.39E+03)	1.0268E+06 (6.34E+03)	1.0278E+06 (6.05E+03)	1.0298E+06 (5.82E+03)
T11.4	1.7882E+04 (1.13E+03)	1.7482E+04 (2.78E+01)	1.7848E+04 (4.24E+01)	1.7850E+04 (8.93E+02)	1.7850E+04 (6.34E+01)
T11.7	1.6832E+06 (4.23E+03)	1.6925E+06 (5.02E+03)	1.7024E+06 (7.82E+03)	1.7024E+06 (8.03E+03)	1.7029E+06 (1.29E+04)
T11.10	9.2368E+05 (6.33E+02)	9.2361E+05 (4.35E+02)	9.2309E+05 (4.27E+02)	9.2380E+05 (5.62E+02)	9.2405E+05 (6.07E+02)
n_{tElite}	Fittest	1Expr-1Expt	Top 30%	Top 35%	Top 40%
T4	1.4572E+01 (5.65E-01)	1.3537E+01 (3.00E-01)	1.4528E+01 (3.73E-01)	1.4697E+01 (3.28E-01)	1.4738E+01 (2.98E-01)
T11.2	1.0277E+06 (7.88E+03)	1.0261E+06 (4.11E+03)	1.0276E+06 (6.46E+03)	1.0269E+06 (1.05E+04)	1.0271E+06 (4.73E+03)
T11.4	1.8256E+04 (5.32E+02)	1.7482E+04 (2.78E+01)	1.7920E+04 (6.72E+02)	1.8125E+04 (8.72E+02)	1.7826E+04 (9.83E+01)
T11.7	1.7189E+06 (7.38E+03)	1.6832E+06 (4.23E+03)	1.7082E+06 (1.12E+04)	1.7196E+06 (5.94E+03)	1.6992E+06 (6.79E+03)
T11.10	9.2516E+05 (6.80E+02)	9.2309E+05 (4.27E+02)	9.2488E+05 (5.39E+02)	9.2513E+05 (6.22E+02)	9.2246E+05 (6.28E+02)

In sensitivity analysis tests, one parameter is varied at a time, and the other parameters are fixed to values as specified in earlier sections. Table IV shows the results of the sensitivity study on the first three parameters (M_{thresh}, $n_{t\text{Elite}}$, and TieR, respectively), while Fig. 8 illustrates the results of the study on varying the window size within which the social fabric is applied and aggregated (Wsize). For each set of parameters, 35 independent runs were performed. As illustrated in Table IV, the mean results and standard deviation for the algorithm on the selected benchmarks were recorded. From this table, it can be noticed that M_{thresh} is the best when set to 50. The table also shows that the effect of varying the tie breaking rules, on the performance of the algorithm is problem dependent. The CA gave the best results when $n_{t\text{Elite}}$ was two: one each from explorer and exploiter KSs in that tribe. On the other hand, Fig. 8 suggests that the best results of the algorithm are obtained when Wsize is 20 for all of the benchmarks save $T_{11.7}$. Hence, Wsize is set at 20 for all test functions.

B. Comparison With Other Algorithms and CA Variants

In order to judge the accuracy of all the algorithms, each algorithm is run until the number of FEs reaches the specified upper limit.

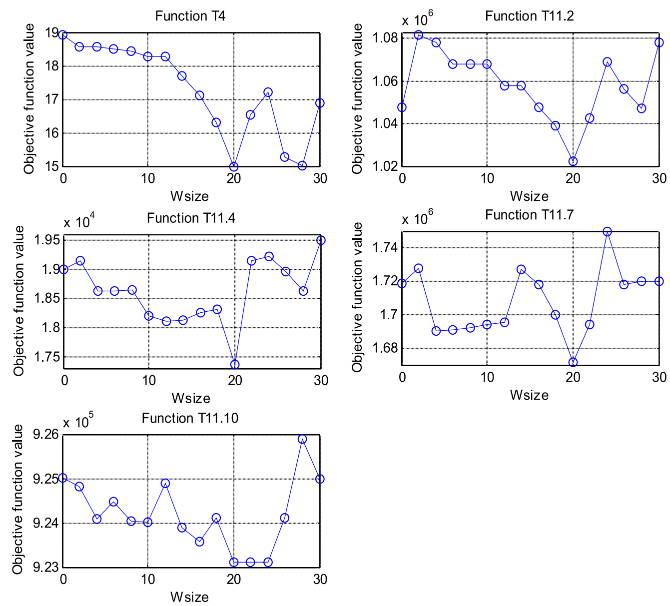

Fig. 8. T-SCANeR results on five selected test functions with different window size Wsize.

In this section, we first compare the performance of T-SCANeR with those of some variants of the tactical restructuring approach using different neighborhood topological schemes, layered-DT [38], and arbitrary rewiring approach (ARA) [38]. The performance will then be compared with five other state-of-the-art algorithms.

1) *DE/rand/1/bin [6]:* A basic differential evolution strategy.
2) *HD-LPSO [42]:* Hierarchical dynamic local neighborhood based PSO.
3) *CA-SF [18]:* CAs with a social fabric metaphor.
4) *MCAKM [14]:* Multipopulation CA adopting knowledge migration.
5) *GA-MPC [43]:* GA with a new multiparent crossover; was the best performing algorithm from the IEEE-CEC2011 competition on real-world optimization problems and is used in this paper also.

A comparison of different variants of T-SCANeR is shown in Table S.X (in supplementary file). The performance of T-SCANeR is assessed when used with tactical restructuring, layered-DT, and with ARA. From Table S.X, it is apparent that the performance is the best when used with tactical restructuring, and gives the worst results when it relies on layered-DT, especially for high-dimensional problems.

For the sake of comparison with the other state-of-the-art algorithms, the best parametric setups for all the EAs and swarm-based techniques were employed [6], [14], [18], [42], as prescribed in their respective sources. In order to save space, the comparisons are limited to the upper limit of 1.5×10^5 for FEs and report the mean and standard deviation of the function values achieved for the benchmark problems in Table V. For a full report on the worst, median, best, mean, and std, please refer to Tables S.XI–S.XIII (in supplementary file).

The simulation results show that T-SCANeR performed at least as well as the other contestant algorithms in terms of

TABLE V
COMPARISON OF PERFORMANCE OF DIFFERENT ALGORITHMS WITH RESPECT TO THE FUNCTION VALUES ACHIEVED FOR TEST PROBLEMS (T1–T13). WITHIN BRACKETS WILCOCOXON'S RANKSUM TEST RESULTS ARE SHOWN WITH 1 (BETTER), 0 (EQUAL), AND −1 (WORSE)

		DE/rand/1/bin	HD-LPSO	CA-SF	MCAKM	GA-MPC	T-SCANeR
T1	Mean	0.000000E+00 (0)	0.000000E+00 (0)	0.000000E+00 (0)	0.000000E+00 (0)	0.000000E+00 (0)	0.000000E+00
	Std	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
T2	Mean	-2.569021E+01 (-1)	-2.548264E+01 (-1)	-2.801342E+01 (-1)	-2.698936E+01 (-1)	-2.720012E+01 (-1)	-3.719333E+01
	Std	8.151509E-01	1.783520E+00	9.999210E-01	5.260856E-01	7.577180E-01	4.092655E-01
T3	Mean	1.408362E-05 (-1)	1.223420E-05 (-1)	1.298541E-05 (-1)	1.089489E-05 (1)	1.151489E-05 (0)	1.151489E-05
	Std	3.973342E-03	4.525479E-04	3.742822E-03	4.090371E-03	0.000000E+00	0.000000E+00
T4	Mean	1.596236E+01 (-1)	1.389798E+01 (1)	1.511010E+01 (-1)	1.747272E+01 (-1)	1.401529E+01 (1)	1.541428E+01
	Std	0.000000E+00	1.345871E-01	1.348707E+00	1.488901E+00	5.728199E-01	9.116182E-01
T5	Mean	-3.098230E+01 (-1)	-2.791272E+01 (-1)	-3.686251E+01 (-1)	-2.488362E+01 (-1)	-3.476852E+01 (-1)	-4.407238E+01
	Std	1.011222E+00	9.991890E-01	6.747201E-01	4.725411E+00	1.213868E+00	4.727017E-01
T6	Mean	-3.167230E+01 (-1)	-3.909310E+01 (1)	-2.542279E+01 (-1)	-2.798858E+01 (-1)	-1.892410E+01 (-1)	-3.825341E+01
	Std	1.572980E+00	1.275626E+00	3.989732E+00	4.253477E+00	1.232582E+00	6.368279E-01
T7	Mean	1.352672E+00 (-1)	6.467139E-01 (-1)	1.182449E+00 (-1)	1.100032E-01 (-1)	9.064066E-01 (-1)	5.726812E-01
	Std	1.290352E+00	2.484320E-01	1.008613E+00	1.128462E+00	2.618831E-01	1.700027E-01
T8	Mean	2.200000E+02 (0)	2.200000E+02 (0)	2.200000E+02 (0)	2.200000E+02 (0)	2.200000E+02 (0)	2.200000E+02
	Std	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00	0.000000E+00
T9	Mean	1.324932E+04 (-1)	1.227812E+03 (-1)	1.188520E+03 (-1)	8.931492E+01 (-1)	2.235051E+03 (-1)	6.163900E+01
	Std	6.893911E+02	3.986251E+02	3.518077E+02	4.843160E+01	2.193687E+03	3.270924E+01
T10	Mean	-1.550200E+01 (-1)	-2.573132E+01 (-1)	-3.224249E+01 (-1)	-2.052631E+01 (-1)	-2.168605E+01 (-1)	-3.780232E+01
	Std	4.349568E+00	2.561810E-01	8.120709E-01	9.277509E-01	1.098801E-01	1.963169E-01
T11.1	Mean	5.936923E+04 (-1)	5.742191E+04 (-1)	5.000297E+04 (-1)	6.233685E+04 (-1)	5.232884E+04 (-1)	4.206010E+04
	Std	5.342091E+02	4.917205E+02	3.879025E+02	0.000000E+00	4.561332E+02	2.201983E+02
T11.2	Mean	1.181629E+06 (-1)	1.165420E+06 (-1)	1.072937E+06 (-1)	1.031767E+06 (-1)	1.083562E+06 (-1)	1.027362E+06
	Std	1.536786E+04	1.101790E+04	3.357892E+03	1.2738810E+03	2.628145E+04	2.472066E+03
T11.3	Mean	1.584238E+04 (-1)	1.544529E+04 (-1)	1.544139E+04 (0)	1.595529E+04 (-1)	1.544436E+04 (-1)	1.543593E+04
	Std	5.820036E+03	1.867130E+02	3.925562E+01	9.937183E+02	8.793188E-01	3.066197E+01
T11.4	Mean	1.907842E+04 (-1)	1.871710E+04 (-1)	1.828732E+04 (-1)	1.913537E+04 (-1)	1.829124E+04 (-1)	1.750196E+04
	Std	9.201787E+01	8.015776E+01	0.000000E+00	8.270833E+01	1.061187E+02	6.032726E+01
T11.5	Mean	3.287122E+04 (-1)	2.284412E+04 (1)	3.331026E+04 (-1)	3.294637E+04 (-1)	3.278424E+04 (-1)	3.139821E+04
	Std	0.000000E+00	3.581763E+01	4.458847E+04	0.000000E+00	3.720831E+01	1.798788E+01
T11.6	Mean	1.298934E+05 (-1)	1.215092E+05 (-1)	1.327898E+05 (-1)	1.215864E+05 (-1)	1.376373E+05 (-1)	1.203198E+05
	Std	1.930341E+03	1.990713E+03	1.101212E+03	1.472411E+03	3.144648E+03	2.216314E+02
T11.7	Mean	1.188835E+07 (-1)	1.978146E+06 (-1)	1.964151E+06 (-1)	1.973221E+06 (-1)	2.337186E+06 (-1)	1.683265E+06
	Std	2.625341E+04	6.123819E+04	5.836458E+04	0.000000E+00	1.207146E+06	4.826176E+03
T11.8	Mean	1.134252E+06 (-1)	9.941871E+05 (-1)	9.379819E+05 (-1)	9.453922E+05 (-1)	1.062854E+06 (-1)	9.186112E+05
	Std	0.000000E+00	1.723172E+04	7.482093E+03	1.231428E+04	2.577344E+05	7.015908E+02
T11.9	Mean	1.296489E+06 (-1)	1.119427E+06 (-1)	1.020034E+06 (-1)	1.010352E+06 (-1)	1.279560E+06 (-1)	9.260034E+05
	Std	4.783972E+04	6.093271E+04	2.438935E+03	4.930476E+04	1.357027E+05	1.134263E+03
T11.10	Mean	9.527892E+05 (-1)	9.642018E+05 (-1)	9.587499E+05 (-1)	9.617275E+05 (-1)	1.022141E+06 (-1)	9.231894E+05
	Std	2.678391E+04	0.000000E+00	1.012034E+04	2.111699E+04	3.895832E+04	1.541782E+02
T12	Mean	1.367813E+01 (-1)	1.107921E+01 (-1)	1.100838E+01 (-1)	1.354165E+01 (-1)	1.372831E+01 (-1)	5.015218E+00
	Std	4.351689E+00	3.136910E+00	3.870237E+00	5.962872E+00	3.257062E+00	1.635382E+00
T13	Mean	1.048978E+01 (-1)	1.141652E+01 (-1)	1.074604E+01 (-1)	1.192552E+01 (-1)	1.218167E+01 (-1)	8.782635E+00
	Std	9.698392E-01	1.895334E+00	1.180423E+00	1.346522E+00	3.343095E+00	7.376327E-01
Ranks		2.6364	3.5455	3.5909	3.5909	3.0227	4.6136
B/E/W		20 / 2 / 0	17 / 2 / 3	19 / 3 / 0	19 / 2 / 1	18 / 3 / 1	

the best and average functional values achieved over 18 functions. It managed to remain second best for function T_3 being outperformed by MCAKM alone. Considering function T_4, T-SCANeR is the third best, where it was outperformed by GA-MPC and HD-LPSO. It also managed to remain second best over functions T_6 and $T_{11.5}$ being outperformed by HD-LPSO alone. T-SCANeR yields results that are better than or comparable to all of the other contestant algorithms on all the remaining cases. Given the results, especially for problems with high dimensionalities and high multimodalities such as T_{09}, $T_{11.1}$, $T_{11.2}$, and $T_{11.7}$–$T_{11.10}$, one can surmise that the average performance for the algorithm was improved due to the emergent social structures in the population space through restructuring. This restructuring helps escape local optima, and yet preserves diversity among the population as described earlier. Problem $T_{11.2}$ has 240 variables and is considered to be a very complex problem, but T-SCANeR was able to obtain a best function value of 1.018614E+06 and an average function value of 1.027362E+06 compared to the corresponding numbers 1.069970E+06 and 1.083562E+06 for the best and

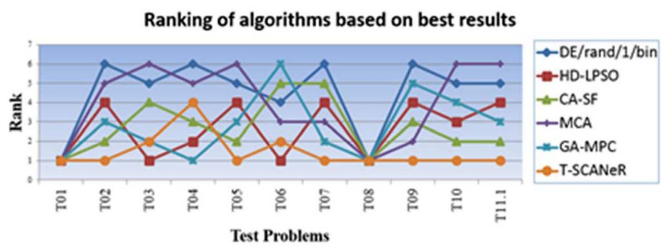

Fig. 9. Ranking of the contestant algorithms shown in Table VI based on the best results over problems T_{01}–$T_{11.1}$. FEs = 1.5×10^5.

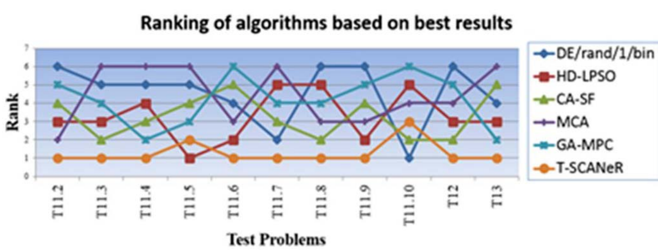

Fig. 10. Ranking of the contestant algorithms shown in Table VI based on the best results over problems $T_{11.2}$–T_{13}. FEs = 1.5×10^5.

average function values obtained by the GA-MPC that won the competition.

Two of the problems from this set are associated with the European Space Agency (ESA). They are the full messenger (T_{12}) and the Cassini problem (T_{13}). The results obtained are very competitive with other algorithms whose results are reported at the ESA. For the messenger problem, T-SCANeR converged with a best value of 2.193724 km/s and an average value of 5.015218 km/s, as compared to 8.663111 km/s, and 13.72831 km/s, for those best and average values obtained by the winner [41]. Here, the T-SCANeR result for this problem is very close to the best reported result on the ESA website as of May 2014 and is reported as 1.959 km/s. For problem T_{13}, T-SCANeR was able to obtain better results than the best in the competition specified in [41]. The best and average results from the approach were 8.383090 and 8.782635 km/s, as compared to 8.609471E and 12.18167 km/s obtained by GA-MPC, which was officially reported as the algorithm with the best results in [41]. The best reported result by MIDACO algorithm [43] is equal to 8.383 km/s, which is similar to that obtained by our approach.

In order to determine the significance of the T-SCANeR CA statistically, a nonparametric statistical test, known as the Wilcoxon's ranksum test [44], is applied on the mean error with a significance level of 5%. The results are presented in Table V inside the parentheses. The numerical values shown in the last row of the table −1, 0, 1 illustrate that the other algorithms are either inferior to, equal to, or superior to the proposed T-SCANeR, respectively.

Table VI reports the adjusted p-values, considering the multiple experiments that were conducted. The tests use Holm's procedure and Shaffer's static procedure [44] in order to acquire adjusted p-values. Numbers marked in boldface indicate a particular case when the hypothesis is rejected at $\alpha = 0.05$.

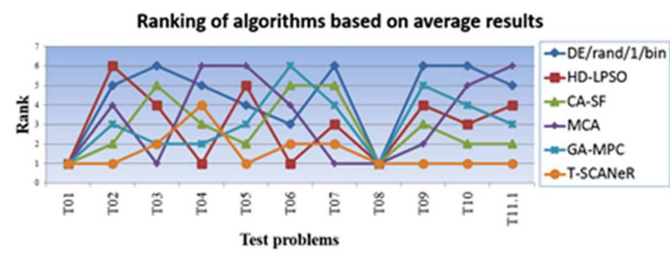

Fig. 11. Ranking of the contestant algorithms shown in Table VI based on the average results over problems T_{01}–$T_{11.1}$. FEs = 1.5×10^5.

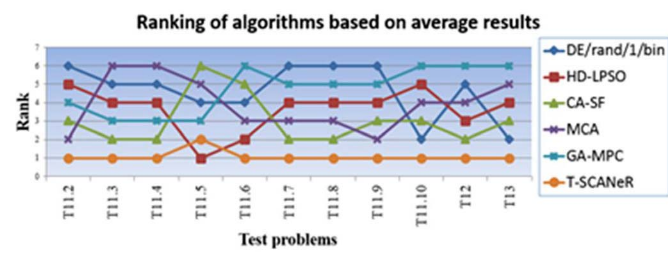

Fig. 12. Ranking of the contestant algorithms shown in Table VI based on the average results over problems $T_{11.2}$–T_{13}. FEs = 1.5×10^5.

TABLE VI
STATISTICAL TEST RESULTS: HOLM/SHAFFER FOR $\alpha = 0.05$

i	algorithms	$z=(R_0-R_i)/SE$	p	*Holm*	*Shaffer*
15	DE vs. T-SCANeR	3.5053	0.0004	**0.0033**	**0.0033**
14	GA-MPC vs. T-SCANeR	2.8203	0.0047	**0.0035**	**0.0050**
13	HD-LPSO vs. T-SCANeR	1.8936	0.0582	**0.0038**	**0.0050**
12	CA-SF vs. T-SCANeR	1.8131	0.0698	**0.0041**	**0.0050**
11	MCAKM vs. T-SCANeR	1.8131	0.0698	**0.0045**	**0.0050**
10	DE vs. CA-SF	1.6922	0.0906	**0.0050**	**0.0050**
9	DE vs. MCAKM	1.6922	0.0906	**0.0055**	**0.0055**
8	DE vs. HD-LPSO	1.6116	0.1070	**0.0062**	**0.0062**
7	CA-SF vs. GA-MPC	1.0072	0.3138	**0.0071**	**0.0071**
6	MCAKM vs. GA-MPC	1.0072	0.3138	**0.0083**	**0.0083**
5	HD-LPSO vs. GA-MPC	0.9266	0.3540	**0.0100**	**0.0100**
4	DE vs. GA-MPC	0.6849	0.4933	**0.0125**	**0.0125**
3	HD-LPSO vs. CA-SF	0.0805	0.9357	**0.0166**	**0.0166**
2	HD-LPSO vs. MCAKM	0.0805	0.9357	**0.0250**	**0.0250**
1	CA-SF vs. MCAKM	0.0000	1.0000	0.0500	0.0500

From these tests, it is clear that there is a statistically significant difference between T-SCANeR and the other algorithms using both statistical test procedures at the chosen level of significance ($\alpha = 0.05$).

Figs. 9 and 10 show the ranking of the algorithms based on the best obtained results, while Figs. 11 and 12 illustrate the ranking of the algorithms based on the average results obtained over the 22 real-world optimization problems. The lower the rank, the better it is where the average rank is given by

$$\text{Average Rank} = \frac{\text{Total Rank}}{44}. \tag{21}$$

Performance comparisons of the contestant algorithms are arranged by ranking the algorithms using Friedman tests [44] according to the mean value that each algorithm obtained for each of the 22 problems, as shown in Table V. The algorithms can be sorted into the following order: T-SCANeR, GA-MPC, MCAKM, CA-SF, HD-LPSO, and DE/rand/1/bin. The best average ranking in these tests was obtained using T-SCANeR which outperformed the other algorithms for this suite of problems.

VI. Conclusion

In this paper, a novel tribal CA with tactical neighborhood restructuring (T-SCANeR) was introduced in order to solve real-world problems of high dimensionality. The idea was to examine the effect of varying the way individuals interact with their neighbors and how it might affect this propagation of information through the networks. The structure of the social fabric connecting the set of problem solvers is dynamically restructured based on the success of the KSs, the degree of interaction of the corresponding bounding boxes of these KSs, and the search mode. Connections among the tribes happen through the two representative types of elites for each tribe, $t\text{Elite}_{\text{Expr}}$, and $t\text{Elite}_{\text{Expt}}$. These connections encourage the propagation of knowledge of the appropriate stream of agents depending on the current search situation.

The performance of T-SCANeR is assessed through a test-suite of real-world benchmarks from the IEEE-CEC2011 EA competition, at different FEs. A comparison with other CA variants and other state-of-the-art algorithms from the literature demonstrates a statistically significant difference in performance of T-SCANeR over the competing algorithms. The results strongly indicate that the neighborhood restructuring was an effective tool in the problem-solving process. Future work will consider other approaches to network restructuring that reflect particular problem structures.

References

[1] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial System*. New York, NY, USA: Oxford Univ. Press, 1999.

[2] J. R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA, USA: MIT Press, 1992.

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA, USA: Addison-Wesley Longman, 1989.

[4] R. G. Reynolds, "An introduction to cultural algorithms," in *Proc. 3rd Annu. Conf. Evol. Program.*, 1994, pp. 131–139.

[5] L. J. Fogel, *Intelligence Through Simulated Evolution: Forty Years of Evolutionary Programming*. New York, NY, USA: Wiley, 1999.

[6] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Global Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[7] J. Kennedy and R. C. Eberhart, *Swarm Intelligence*. San Francisco, CA, USA: Morgan Kaufmann, 2001.

[8] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, May 1983.

[9] S. S. Haykin, *Neural Networks: A Comprehensive Foundation*. Upper Saddle River, NJ, USA: Prentice Hall, 1999.

[10] L. N. de Castro and J. Timmis, *Artificial Immune Systems: A New Computational Intelligence Approach*. New York, NY, USA: Springer, 2002.

[11] R. G. Reynolds and M. Z. Ali, "The social fabric approach as an approach to knowledge integration in cultural algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Hong Kong, Jun. 2008, pp. 4200–4207.

[12] C.-J. Chung and R. G. Reynolds, "CAEP: An evolution-based tool for real-valued function optimization using cultural algorithms," *Int. J. Artif. Intell. Tools*, vol. 7, no. 3, pp. 239–291, Sep. 1998.

[13] B. Peng, "Knowledge and population swarms in cultural algorithms for dynamic environments," Ph.D. dissertation, Dept. Comp. Sci., Wayne State Univ., Detroit, MI, USA, 2005.

[14] Y.-N. Guo, J. Cheng, Y.-Y. Cao, and Y. Lin, "A novel multi-population cultural algorithm adopting knowledge migration," *Soft Comput.*, vol. 15, no. 5, pp. 897–905, May 2011.

[15] R. G. Reynolds and B. Peng, "Cultural algorithms: Modeling of how cultures learn to solve problems," in *Proc. 16th Int. Conf. Tools Artif. Intell. (ICTAI)*, Boca Raton, FL, USA, Nov. 2004, pp. 166–172.

[16] M. Z. Ali, A. M. Salhieh, and R. G. Reynolds, "Socio-cultural evolution via neighborhood-restructuring in intricate multi-layered networks," in *Proc. IEEE Congr. Evol. Comput.*, Brisbane, QLD, Australia, Jun. 2012, pp. 1–8.

[17] M. Z. Ali, R. G. Reynolds, and R. Ali, "Enhancing cultural learning under environmental variability using layered heterogeneous sociometry-based networks," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Intell. Agent Technol.*, Toronto, ON, Canada, Aug. 2010, pp. 69–74.

[18] C. Xiangdong, M. Ali, and R. G. Reynolds, "Robust evolution optimization at the edge of chaos: Commercialization of culture algorithms," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[19] R. G. Reynolds, M. Ali, and T. Jayyousi, "Mining the social fabric of archaic urban centers with cultural algorithms," *Computer*, vol. 41, no. 1, pp. 64–72, Jan. 2008.

[20] M. Z. Ali, A. Salhieh, R. T. A. Snanieh, and R. G. Reynolds, "Boosting cultural algorithms with an incongruous layered social fabric influence function," in *Proc. IEEE Congr. Evol. Comput.*, New Orleans, LA, USA, Jun. 2011, pp. 1225–1232.

[21] M. Z. Ali and R. G. Reynolds, "Cultural algorithms—A tabu search approach for the optimization of engineering design problems," *Soft Comput.*, vol. 18, no. 8, pp. 1631–1644, Nov. 2013.

[22] M. Z. Ali, K. Al-Khatib, and Y. Tahstoush, "Cultural algorithms: Emerging social structures for the solution of complex optimization problems," *Int. J. Artif. Intell.*, vol. 11, no. A13, pp. 20–42, Oct. 2013.

[23] W. A. Brock and S. N. Durlauf, "Discrete choice with social interactions," *Rev. Econ. Stud.*, vol. 68, no. 2, pp. 235–260, Apr. 2001.

[24] K. Chen, T. Li, and T. Cao, "Tribe-PSO: A novel global optimization algorithm and its application in molecular docking," *Chemometr. Intell. Lab. Syst.*, vol. 82, nos. 1–2, pp. 248–259, May 2006.

[25] J. Kennedy, "Small worlds and mega-minds: Effects of neighborhood topology on particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3. Washington, DC, USA, 1999, pp. 1931–1938.

[26] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," in *Proc. IEEE Congr. Evol. Comput.*, Honolulu, HI, USA, 2002, pp. 1671–1676.

[27] S. Janson and M. Middendorf, "A hierarchical particle swarm optimizer and its adaptive variant," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 35, no. 6, pp. 1272–1282, Dec. 2005.

[28] A. Mohais, R. Mendes, C. Ward, and C. Postoff, "Neighborhood restructuring in particle swarm optimization," in *Proc. 18th Australian Joint Conf. Artif. Intell.*, Sydney, NSW, Australia, Dec. 2005, pp. 776–785.

[29] S. B. Akat and V. Gazi, "Particle swarm optimization with dynamic neighborhood topology: Three neighborhood strategies and preliminary results," in *Proc. IEEE Swarm Intell. Symp.*, St. Louis, MO, USA, Sep. 2008, pp. 1–8.

[30] P. N. Suganthan, "Particle swarm optimiser with neighbourhood operator," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3. Washington, DC, USA, 1999, p. 1958–1962.

[31] R. Mendes, J. Kennedy, and J. Neves, "The fully informed particle swarm: Simple, maybe better," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 204–210, Jun. 2004.

[32] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[33] S. Z. Zhao, P. N. Suganthan, and S. Das, "Dynamic multi-swarm particle swarm optimizer with sub-regional harmony search," in *Proc. IEEE Congr. Evol. Comput.*, Barcelona, Spain, Jul. 2010, pp. 1–8.

[34] T. Blackwell and J. Branke, "Multi-swarm optimization in dynamic environments," in *Applications of Evolutionary Computing*, vol. 3005, G. Raidl, *et al.*, Eds. Berlin, Germany: Springer, 2004, pp. 489–500.

[35] C. D. Chio, P. D. Chio, and M. Giacobini, "An evolutionary game-theoretical approach to particle swarm optimisation," in *Proc. Conf. Appl. Evol. Comput.*, Naples, Italy, Mar. 2008, pp. 575–584.

[36] A. Silva, A. Neves, and E. Costa, "An empirical comparison of particle swarm and predator prey optimisation," in *Artificial Intelligence and Cognitive Science* (LNCS 2464), M. O'Neill, *et al.*, Eds. Berlin, Germany: Springer, pp. 1–45.

[37] T. Peram, K. Veeramachaneni, and C. K. Mohan, "Fitness-distance-ratio based particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Indianapolis, IN, USA, Apr. 2003, pp. 174–181.

[38] R. Apu and M. Gavrilova, "Efficient swarm neighborhood management using the layered Delaunay triangulation," in *Generalized Voronoi Diagram: A Geometry-Based Approach to Computational Intelligence*, vol. 158. M. Gavrilova Ed. Berlin, Germany: Springer, 2009, pp. 109–129.

[39] X. Cai, Z. Cui, J. Zeng, and Y. Tan, "Dispersed particle swarm optimization," *Inform. Process. Lett.*, vol. 105, no. 6, pp. 231–235, Mar. 2008.

[40] S. E. Sterling, "Aggregation techniques to characterize social networks," master's thesis, School Eng. Manage., Storming Media, Air Force Instit. Technol., Wright-Patterson AFB, OH, USA, 2004.

[41] S. Das and P. N. Suganthan, "Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems," Dept. Electron. Telecommun. Eng., Jadavpur Univ., Kolkata, India, Nanyang Technological Univ., Singapore, Tech. Rep., Dec. 2010.

[42] P. Ghosh, H. Zafar, S. Das, and A. Abraham, "Hierarchical dynamic neighborhood based particle swarm optimization for global optimization," in *Proc. IEEE Congr. Evol. Comput.*, New Orleans, LA, USA, Jun. 2011, pp. 757–764.

[43] S. M. Elsayed, R. A. Sarker, and D. Essam, "GA with a new multi-parent crossover for constrained optimization," in *Proc. IEEE Congr. Evol. Comput.*, New Orleans, LA, USA, Jun. 2011, pp. 857–864.

[44] S. Garcia, A. Fernandez, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining," *Inf. Sci.*, vol. 180, no. 10, pp. 2044–2064, May 2010.

Mostafa Z. Ali (SM'15) received the bachelor's degree in applied mathematics from Jordan University of Science and Technology, Irbid, Jordan, in 2000; the master's degree in computer science from University of Michigan–Dearborn, Dearborn, MI, USA, in 2003; and the Ph.D. degree in computer science/artificial intelligence from Wayne State University, Detroit, MI, USA, in 2008.

He is an Associate Professor with the Department of Computer Information Systems, Jordan University of Science and Technology. His research interests include evolutionary computation, cultural algorithms, virtual reality, data mining, and bioinformatics databases.

Dr. Ali is an Associate Editor of *Swarm and Evolutionary Computation* (Elsevier). He is a member of the American Association of Artificial Intelligence and Association for Computing Machinery.

Ponnuthurai N. Suganthan (S'91–M'92–SM'00–F'15) received the B.A. degree, the Postgraduate Certificate, and the M.A. degree in electrical and information engineering from University of Cambridge, Cambridge, U.K., in 1990, 1992, and 1994, respectively, and the Ph.D. degree from the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore.

He is an Associate Professor with the School of Electrical and Electronic Engineering, Nanyang Technological University. His research interests include evolutionary computation, pattern recognition, multiobjective evolutionary algorithms, bioinformatics, and neural networks.

Dr. Suganthan is an Associate Editor of IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, *Information Sciences*, *Pattern Recognition*, and *International Journal of Swarm Intelligence Research*. He is a Founding Co-Editor-in-Chief of *Swarm and Evolutionary Computation* (Elsevier).

Robert G. Reynolds (SM'13) received the bachelor's degree from University of Michigan-Dearborn, Dearborn, MI, USA, in 1970, and the master's and Ph.D. degrees from University of Michigan, Ann Arbor, MI, USA, in 1978 and 1979, respectively, all in computer science.

He is a Professor of Computer Science with Wayne State University, Detroit, MI, USA, and an Associate Research Scientist of the Museum of Anthropology with University of Michigan. His research interests include artificial intelligence, evolutionary computation, and cultural algorithms.

Dr. Reynolds is a member of Association for Computing Machinery, the American Association of Artificial Intelligence, the American Association for the Advancement of Science, and the Society for American Archaeology.

Amer F. Al-Badarneh received the B.Sc. degree from Yarmouk University, Irbid, Jordan, in 1987, and the M.Sc. and Ph.D. degrees from Wayne State University, Detroit, MI, USA, in 1995 and 1999, respectively, all in computer science.

He is an Associate Professor of Computer Science with Jordan University of Science and Technology, Irbid. His research interests include data mining, natural language processing, spatial data indexing, and evolutionary computation. He has co-authored several research articles in refereed conferences and journals.