Article:
Cao, B., Zhao, J., Yang, P. orcid.org/0000-0002-8553-7127 et al. (6 more authors) (2018) Distributed parallel cooperative coevolutionary multi-objective large-scale immune algorithm for deployment of wireless sensor networks. Future Generation Computer Systems, 82. pp. 256-267. ISSN 0167-739X

https://doi.org/10.1016/j.future.2017.10.015

Distributed Parallel Cooperative Coevolutionary Multi-Objective Large-Scale Immune Algorithm for Deployment of Wireless Sensor Networks

Bin Cao[a,b,c,*], Jianwei Zhao[a,b,c], Po Yang[d,*], Zhihan Lv[e], Xin Liu[f], Xinyuan Kang[a,b,c], Shan Yang[a,b,c], Kai Kang[f], Amjad Anvari-Moghaddam[g]

[a]*School of Computer Science and Engineering, Hebei University of Technology, 300401 Tianjin, China*
[b]*Key Laboratory of Machine Intelligence and Advanced Computing, Ministry of Education, 510006 Guangzhou, China*
[c]*Hebei Provincial Key Laboratory of Big Data Calculation, 300401 Tianjin, China*
[d]*Department of Computer Science, Liverpool John Moores University, Liverpool, UK*
[e]*School of Data Science and Software Engineering, Qingdao University, 266071 Qingdao, China.*
[f]*Hebei University of Technology, 300401 Tianjin, China.*
[g]*Department of Energy Technology, Power Electronic Systems, 9220 Aalborg, Denmark.*

Abstract

The use of immune algorithms is generally a time-intensive process—especially for problems with numerous variables. In the present paper, we put forward a distributed parallel cooperative coevolutionary multi-objective large-scale immune algorithm that is implemented using the message passing interface (MPI). The proposed algorithm comprises three layers: objective, group and individual layers. First, to address each objective in a multi-objective problem, a subpopulation is used for optimization, and an archive population is used to optimize all the objectives. Second, the numerous variables are divided into several groups. Finally, individual evaluations are allocated across many core processing units, and calculations are performed in parallel. Consequently, the computation time is greatly reduced. The proposed algorithm integrates the idea of immune algorithms, which explore sparse areas in the objective space, and uses simulated binary crossover for mutation. The proposed algorithm is employed to optimize the 3D terrain deployment of a wireless sensor network, which is a self-organization network. In our experiments, through comparisons with several state-of-the-art multi-objective evolutionary algorithms—the cooperative coevolutionary generalized differential evolution 3, the cooperative multi-objective differential evolution, the multi-objective evolutionary algorithm based on decision variable analyses and the nondominated sorting genetic algorithm III—the proposed algorithm addresses the deployment optimization problem efficiently and effectively.

Keywords: decision variable analysis (DVA), cooperative coevolution (CC), large-scale optimization, message passing interface (MPI), wireless sensor networks (WSNs), 3D terrain deployment

1. Introduction

In the wireless sensor network (WSN) deployment optimization procedure [1], wireless sensor nodes can be optimized via self-organization [2] to maximize the *Coverage*, optimize the *Connectivity Uniformity* and minimize the *Deployment Cost*. With the rapid development of sensor and wireless communication technologies, WSNs have been applied to various fields. The work of [3] presented an air temperature monitoring application for WSNs. Shen et al. [4] described the wireless sensor nodes for a medical service. Zhang et al. [5] illustrated the WSN k-barrier coverage problem. Zhou et al. [6] researched the energy issue, regarding which clustering and data compression were studied. Zhang et al. [7] utilized mobile sinks to alleviate the communication burden.

In addition, the response of the human immune system to antigens can be viewed as a process of self-organization. Based on this concept, the clonal selection algorithm (CLONALG) [8], which can be used for global optimization problems (GOPs) and multi-objective optimization problems (MOPs) [9], was proposed. Other nature-inspired algorithms also follow the self-organizing procedure. For example, Xue et al. [10] described the self-adaptive artificial bee colony algorithm, which is different from the immune algorithm.

In the real world, many problems require several (usually conflicting) objectives to be considered simultaneously. Multi-objective evolutionary algorithms (MOEAs) [11, 12, 13] are capable of producing a plurality of solutions during one run, which is convenient for approximating the Pareto front (PF). For NP-hard problems, evolutionary algorithms (EAs) [14, 15, 16, 17] can usually converge to near-optimal solutions using limited computational resources [18] within a reasonable time compared to brute force and deterministic methods.

The first multi-objective immune algorithm (MOIA) was proposed in [19]. In this study, the immune algorithm (IA) was integrated into the genetic algorithm (GA) to improve the selection of individuals for evolution. Gong et al. [20] presented the nondominated neighbor immune algorithm (NNIA), which se-

*Corresponding authors.
E-mail addresses: p.yang@ljmu.ac.uk (Po Yang), caobin@scse.hebut.edu.cn (Bin Cao)

lects a small quantity of nondominated individuals in a sparse area for cloning, recombination and mutation. In [21], simulated binary crossover (SBX) and differential evolution (DE) were combined and applied to cloned individuals in a hybrid evolutionary framework for MOIAs called HEIA, which performed well for both unimodal and multimodal problems.

EAs are based on an iterative evolution of the population (the solutions), which is time-consuming—especially for expensive problems. Distributed evolutionary algorithms (dEAs) [22, 23] allocate the tedious computational burden across numerous computational nodes, greatly reducing the required time. Cloudde [24] used DEs with various parameters to optimize multiple populations in a distributed parallel manner, yielding a promising performance from both the effect and efficiency aspects. [25] provided a comprehensive study concerning parallel/distributed MOEAs. Utilizing the multi-objective optimization algorithm based on decomposition (MOEA/D) [13], parallel MOEA/Ds (pMOEA/Ds) [26] [27] were proposed.

With the arrival of "big data", many complex problems have emerged; solving such problems is both time-consuming and storage-consuming [28, 29]. Similarly, many MOPs now have numerous variables (e.g., more than 100 variables [30]). Some examples include classification [31], clustering [32], and recommendation systems [33]. However, the goal of traditional MOEAs is to solve multi-objective small-scale optimization problems (MOSSOPs). Consequently, the traditional algorithms may be incapable of tackling multi-objective large-scale optimization problems (MOLSOPs) because of the "curse of dimensionality". To optimize numerous variables, some promising approaches first separate the variables into groups and then optimize them in a cooperative coevolutionary (CC) [34] manner. For large-scale global optimization problems (LSGOPs), many grouping mechanisms have been applied, including fixed grouping [34], random grouping [35], the Delta method [36], dynamic grouping [37], differential grouping (DG) [38], global differential grouping (GDG) [39] and graph-based differential grouping (gDG) [40]. Antonio et al. proposed the cooperative coevolutionary generalized differential evolution 3 (CCGDE3) method [41], which used fixed grouping.

MOLSOPs differ from LSGOPs in that no single solution optimizes all the conflicting objectives; instead, a solution set should be generated to approximate the PF. In MOLSOPs, variables have different properties [42], which can be classified as follows:

1. position variables, which affect only the diversity of the solution set;
2. distance variables, which affect only the convergence of the solution set; and
3. mixed variables, which affect both the diversity and the convergence of the solution set.

Therefore, position variables should be permuted to approximate the PF as comprehensively as possible. However, distance variables should be optimized so that they can closely approach the PF.

To identify these variable types, the multi-objective evolutionary algorithm based on decision variable analyses (MOEA/DVA) [30] utilizes a mechanism called decision variable analyses (DVA). The position as well as mixed variables are categorized as diversity-related variables, while distance variables, as convergence-related variables. The convergence-related variables are allocated to multiple groups that are then optimized under the CC framework.

The use of multiple populations can impact the optimization performance. In cooperative multi-objective differential evolution (CMODE) [43], each objective is optimized by a subpopulation, and an archive is used to maintain good solutions and optimize all objectives. This approach has yielded good experimental results.

Compared to MOSSOPs, designing parallel/distributed MOEAs for MOLSOPs will be more beneficial. In this paper, we propose the distributed parallel cooperative coevolutionary multi-objective large-scale immune algorithm (DPCC-MOLSIA), which is aimed at solving MOLSOPs effectively and efficiently.

The contributions of this paper can be summarized as follows:

1. Each objective is optimized by a subpopulation. Thus, the exploration with respect to each objective is enhanced, and all objectives are comprehensively optimized by an archive. Variables are grouped according to their properties and interactions, contributing to effective optimization.
2. The idea of the IA is introduced, more computational resources are used to explore sparse areas in the objective space, and *SBX* is utilized for evolution.
3. We construct a three-layer parallel structure. The evaluations of individuals in different groups of multiple populations can then be performed in parallel, which greatly reduces the computation time.

The remainder of this paper is organized as follows: Section 2 provides some preliminary information required for this paper. The details of the DPCCMOLSIA are discussed in Section 3. Then, in Section 4, we describe the experimental study and present the corresponding analyses. Finally, Section 5 concludes this paper.

2. Preliminaries

2.1. MOP and Variable Properties

An MOP involves several objectives that usually conflict with each other. Therefore, addressing an MOP comprises obtaining a solution set that approximates the PF. For the minimization problem, we have the following formula:

$$\text{Minimize } F(\boldsymbol{X}) = \{f_1(\boldsymbol{X}), f_2(\boldsymbol{X}), ..., f_M(\boldsymbol{X})\}, \quad (1)$$

where $\boldsymbol{X} = (X_1, X_2, ..., X_D)$ is a point in the solution space $\Re^D$. Here, D denotes the variable quantity, f_i, $i = 1, 2, ..., M$, represents the objectives, and $F(\boldsymbol{X})$ denotes the point in the objective space $\Re^M$ that corresponds to $\boldsymbol{X}$.

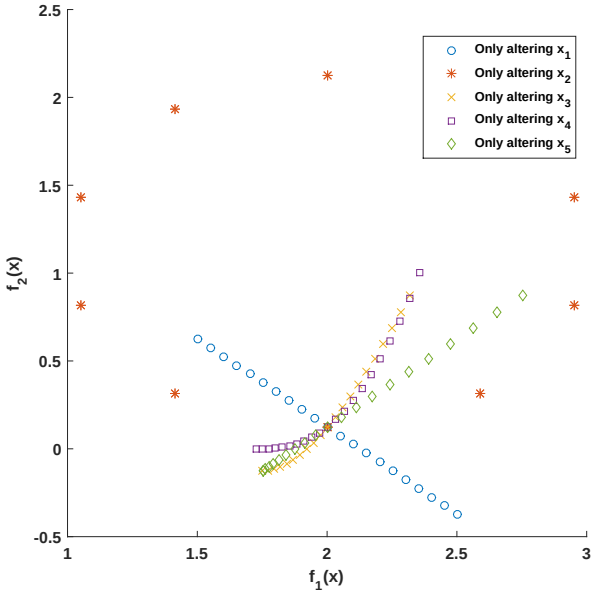


Figure 1: Image of solution sets for the MOP formulated in Eq. 2, sampled by altering one variable while holding the others constant at 0.5.

Due to the conflicts among objectives, the types of different variables involved can be numerous; correspondingly, variables can be classified as position, distance, and mixed variables. For instance, consider the following MOP:

$$\begin{cases} f_1 = 0 + x_1 + \sin(4\pi x_2) + e^{x_3(x_4-0.05)} + x_5^2 \\ f_2 = 1 - x_1 - \cos(4\pi x_2) + x_3^2 + x_4^3 + x_5^2 \\ \quad \text{s.t. } x_i \in [0,1],\ i = 1,2,3,4,5, \end{cases} \tag{2}$$

where f_1 and f_2 are two objectives and x_1, x_2, x_3, x_4 and x_5 are decision variables.

Fig. 1 illustrates the solution sets sampled by altering each variable individually while holding the others constant at 0.5. From the image, we can determine the properties of the variables: x_1 is a position variable, as it influences only the diversity; x_2 is a mixed variable, as it influences both the diversity and the convergence; x_3 and x_4 are distance variables, though their relative positions change only slightly with variation of the values; and x_5 is a distance variable, as it influences only the convergence.

2.2. CC

CC [34] divides a great quantity of variables into several subcomponents that are optimized separately. For fitness evaluation, the target subcomponent is recombined with representatives from the other components to constitute a complete solution.

2.3. Immune Algorithm

The CLONALG was proposed in [8]; its process is detailed in Algorithm 1. In the CLONALG, an *antibody* denotes a candidate solution, the optimal solution is seen as the *antigen*, and the *affinity* represents the fitness.

Algorithm 1: CLONALG

Input: number of variables: D;
number of antibodies: N_{Ab};
number of generations: N_{gen};
antibodies: POP_{Ab};
number of antibodies to be selected: N_{sel}.
Output: final antibodies: POP_{Ab};
final affinities: AFF_{Ab}.

/* Initialization */
$G = 0$;
Randomly initialize POP_{Ab}, $AFF_{Ab} = f(POP_{Ab})$;
Selected antibodies $POP_{sel} = \phi$, $AFF_{sel} = \phi$;
Reproduced antibodies $POP_{rep} = \phi$, $AFF_{rep} = \phi$;
/* Main Loop */
while $G < N_{gen}$ **do**
  Selection according to AFF_{Ab}^G:
  $POP_{Ab}^G \rightarrow POP_{sel}^G$, $AFF_{Ab}^G \rightarrow AFF_{sel}^G$;
  Cloning according to AFF_{sel}^G:
  $POP_{sel}^G \rightarrow POP_{rep}^G$;
  Hypermutation:
  $POP_{rep}^G \rightarrow \widetilde{POP}_{rep}^G$, $\widetilde{AFF}_{rep}^G = f\left(\widetilde{POP}_{rep}^G\right)$;
  Insertion:
  $POP_{Ab}^G + \widetilde{POP}_{rep}^G \rightarrow POP_{Ab}^{G+1}$, $AFF_{Ab}^{G+1} = f\left(POP_{Ab}^{G+1}\right)$;
  $G++$;

Algorithm 2: DPCCMOLSIA

Initialization;
Variable property and interaction analyses;
Variable grouping;
Parallelism implementation;
Optimization;

3. The Proposed Algorithm: DPCCMOLSIA

Algorithm 2 lists the main steps in the framework of the DPCCMOLSIA. The main procedure is detailed in the following subsections.

3.1. Variable Property and Interaction Analyses

Variables are classified as position variables, distance variables and mixed variables according to their influences on diversity and convergence. At the end of this process, the position variables and mixed variables are categorized as diversity-related variables, and the distance variables are categorized as convergence-related variables. For the MOP formulated in Eq. 2, x_1 and x_2 are classified as diversity-related variables, while x_3, x_4 and x_5 are classified as convergence-related variables.

3.2. Variable Grouping

Because more than one objective exists, the interactions among variables are obtained with respect to each objective by adopting the idea of gDG [40]. The diversity-related variables are separated into a single group. We group the convergence-related variables according to the following idea: if two variables interact with each other for any objective optimized in the present subpopulation/archive, we consider them to be interacting. Take the MOP mentioned above in Eq. 2 for example, x_1 and x_2 are diversity-related variables; thus, they are grouped together. For the convergence-related variables, x_3 and x_4 interact in f_1 and act independently in f_2; thus, we allocate them to a single group in subpopulation 1 (only optimizing f_1), to separate groups in subpopulation 2 (only optimizing f_2), and to the same group in the archive (optimizing both f_1 and f_2). x_5 is independent from other variables for both f_1 and f_2; thus, it is allocated to another separate group.

3.3. Parallelism Implementation

For MOLSOPs, especially expensive ones, parallelism can be beneficial. The DPCCMOLSIA is a distributed parallel algorithm implemented using the MPI. In the DPCCMOLSIA, the parallel structure has three layers.

Assuming that N^{CPU} CPU resources are available, the variables are divided into N_i^G groups. Here, $i = 1, 2, ..., M + 1$—the subpopulations are represented by $i = 1, 2, ..., M$, and the archive is represented by $i = M + 1$. NP individuals exist in each subpopulation and in the archive population. Then, we have the following equation:

$$N_i^{CPU} = \frac{N_i^G}{\sum_{j=1}^{M+1} N_j^G} \times N^{CPU} \quad \text{s.t. } i = 1, 2, ..., M + 1, \tag{3}$$

where N_i^{CPU} denotes the quantity of CPUs allocated to the subpopulation i or the archive.

$$N_{i,j}^{CPU} = \frac{N_i^{CPU}}{N_i^G} \quad \text{s.t. } j = 1, 2, ..., N_i^G, \tag{4}$$

where $N_{i,j}^{CPU}$ is the quantity of CPUs in the charge of group j in subpopulation i or the archive.

The evaluations of the individuals are allocated across the multiple CPUs in each group.

$$N_{i,j,k}^{CPU} = \frac{NP}{N_{i,j}^{CPU}} \quad \text{s.t. } k = 1, 2, ..., N_{i,j}^{CPU}, \tag{5}$$

where $N_{i,j,k}^{CPU}$ is the number of individuals that are assigned to CPU k of group j in subpopulation i or the archive.

Therefore, based on the three-layer parallel structure, the evaluations of the individuals in each group of all $M + 1$ populations are conducted in parallel, which substantially reduces the computation time.

To guarantee the optimization performance, information must be shared among the groups. Hence, the communication strategy should be properly designed [44, 45]; we adopt the von Neumann topology.

Algorithm 3: Evolution

Input: generation number: N_{gen}.
Output: final population: POP_{final}.
1 **for** $G = 1 \rightarrow N_{gen}$ **do**
2 Evolve all variable groups in the subpopulations (Algorithm 4) and the archive (Algorithm 5) in parallel;
3 Exchange information among the groups;
4 Gather all the individuals from all groups to generate the final population POP_{final};

3.4. Evolution Combined with the Idea of the IA

The overall evolution process is provided by Algorithm 3. The evolution of each group in the subpopulations (Algorithm 4) or in the archive (Algorithm 5) is described in the following subsections.

3.4.1. Subpopulations

In Line 2 of Algorithm 4, in the evolution, tour selection is employed to choose 2 individuals from the full population. Then, in Lines 3 and 4, we use *SBX* to evolve variables in the target group and integrate them with other variables to form a complete individual.

$$\widetilde{X}_{i,j} = \begin{cases} SBX(X_i, X_{r_1}, X_{r_2}, j) & \text{if } j \in index \\ X_{r_3,j} & \text{otherwise,} \end{cases} \tag{6}$$

where $\widetilde{X}_i$ denotes the generated new solution, X_i is the target parent individual, X_{r_1} and X_{r_2} are the 2 reference individuals,

Algorithm 4: Evolution of One Variable Group in Subpopulations

Input: number of individuals: NP;
 population: POP_1.
Output: new population: POP_{new1}.
`/* Evolution */`
1 **for** $i = 1 \rightarrow NP$ **do**
2 Select 2 reference individuals;
3 Use *SBX* to generate offspring i;
4 Combine the generated offspring with other variables to construct a complete solution;
5 Perform *polynomial mutation*;
`/* Evaluation */`
6 Allocate the generated solutions to the CPU resources in the group and perform the evaluations in the CPUs in parallel;
7 Collect the fitness values from the CPUs;
`/* Refinement */`
8 Combine the generated solutions with the old population;
9 Obtain NP individuals with respect to their fitness values of the considered objective$\rightarrow$ POP_{new1};

index contains the variables to be optimized by the present group, and X_{r_3} is integrated with the optimized variables to form a complete solution, which has the following form:

$$r_3 = \begin{cases} i & \text{if } r < \dfrac{G}{N_{gen}} \\ r_4 & \text{else if } r' < 0.5 \\ r_5 & \text{otherwise,} \end{cases} \tag{7}$$

where G denotes the present generation number and N_{gen} denotes the maximum generation number. Here, r and r' are random numbers generated uniformly within [0.0, 1.0], and r_4 and r_5 are two individuals selected via tour selection. Then, in Line 5, *polynomial mutation* is performed.

In Lines 6 and 7, to evaluate the newly generated solutions, we use parallelism to alleviate the computational burden. This is the third layer of the parallel structure of the DPCCMOLSIA.

Finally, in Lines 8 and 9, the NP best individuals with respect to the considered objective are preserved.

3.4.2. Archive

Traditionally, in each generation, all individuals take part in evolution. However, this paper introduces the idea of the IA, in which, in each generation, we select several good individuals and produce NP offspring, the entire process of which is illustrated in Algorithm 5. In detail, the selection of individuals in Line 1 is determined by two criteria: nondominance and crowding distance. If the quantity of nondominated individuals is less than N_{sel}, all of them are selected for cloning; otherwise, we select the N_{sel} individuals that have larger crowding distances. In the cloning process in Line 2, the quantity of replicates of each selected individual depends on the crowding distance [21].

Algorithm 5: Evolution of One Variable Group in Archive

Input: number of individuals: NP;
population: POP_2;
maximum number of individuals to be selected: N_{sel}.
Output: new population: POP_{new2}.
`/* Selection */`
1 Select N_{sel} individuals according to the Pareto dominance and crowding distance;
`/* Clone */`
2 Clone the selected individuals to a total number of NP;
`/* Evolution */`
3 **for** $i = 1 \rightarrow NP$ **do**
4 Select 2 reference individuals;
5 Use *SBX* to generate the offspring i;
6 Combine the generated offspring with other variables to construct a complete solution;
7 Perform *polynomial mutation*;
`/* Evaluation */`
8 Allocate the generated solutions to the CPU resources in the group and perform evaluations on the CPUs in parallel;
9 Collect the fitness values from the CPUs;
`/* Nondominated sorting */`
10 Combine the generated solutions with the old population;
11 Obtain NP individuals according to the Pareto dominance and crowding distance→ POP_{new2};

$$N_i^C = \frac{dist_i}{\sum_{j=1}^{N_{sel}} dist_j} \times NP, \tag{8}$$

where N_i^C represents the number of replications of selected individual i and $dist_i$ is its crowding distance in the population, which is calculated as follows:

$$dist_i = \sum_{m=1}^{M} dist_i^m, \tag{9}$$

where $dist_i^m$ denotes the crowding distance of the i-th individual with respect to objective m, with

$$dist_i^m = \begin{cases} \infty & \text{if } (i)^* = 1 \\ 0 & \text{if } (i)^* = NP \\ \dfrac{\widetilde{f}_m^{(i)^*+1} - \widetilde{f}_m^{(i)^*-1}}{\widetilde{f}_m^{NP} - \widetilde{f}_m^{1}} & \text{otherwise.} \end{cases} \tag{10}$$

$\widetilde{f}_m^{(i)^*}$ is the f_m^i sorted in descending order, and $(i)^*$ denotes the new index of the i-th individual in the sorted sequence.

$$dist_i = \begin{cases} 2 \times dist_i^{max} & \text{if } dist_i = \infty \\ dist_i & \text{otherwise,} \end{cases} \tag{11}$$

where $dist_i^{max}$ is the maximum crowding distance. Because ∞ values are assigned to the crowding distances, to calculate N_i^C, we have to convert them.

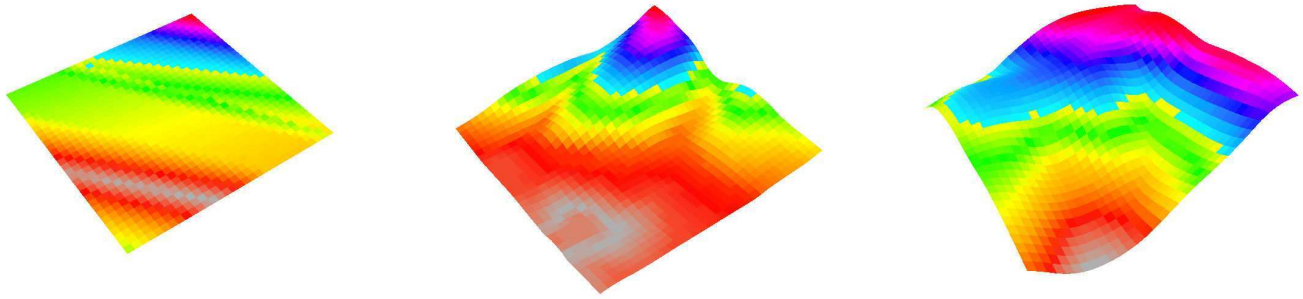
In Line 4 of the evolution process, we select 2 individuals from among the N_{sel} selected individuals if $N_{sel} > 2$; otherwise, the selection scope is the whole population. Then, in Lines 5 and 6, we use *SBX* to generate the target individual. For the integration, r_4 and r_5 (Eq. 7) are 2 randomly selected individuals from the N_{sel} best individuals used for cloning when $N_{sel} > 2$ or from the whole population when $N_{sel} \leq 2$. Then, in Line 7, *polynomial mutation* is performed.

Finally, in Lines 10 and 11, we combine the new individuals with the present population to obtain the NP best individuals according to the Pareto dominance and crowding distance. When the quantity of nondominated individuals is below NP, several dominated individuals will be preserved.

4. Experimental Research: Application to 3D Terrain Deployment of Heterogeneous Directional Sensor Networks

4.1. 3D Deployment Problem and Terrain Data

We use the 3D deployment problem proposed in [1], which includes three objectives: *Coverage*, *Connectivity Uniformity* and *Deployment Cost*. We also utilize the same real-world 3D terrain data (Fig. 2), which are composed of plain (Fig. 2a), hilly (Fig. 2b) and mountainous (Fig. 2c) terrains. These three terrains have different characteristics that are used to verify the proposed algorithm with respect to various conditions.



(a) Plain Terrain (b) Hilly Terrain (c) Mountainous Terrain

Figure 2: Illustration of 3D terrain data.

4.2. Parameter Setup

We compare the DPCCMOLSIA with the CCGDE3 [41], the CMODE [43], the MOEA/DVA [30] and the nondominated sorting genetic algorithm III (NSGA-III) [46] in terms of addressing the deployment optimization problem.

For all the algorithms, the optimization process is performed 24 times. The fitness evaluations (FEs) are set to $10^4 \times D$; here, $D = 2 \times 10^2$.

To ensure fair comparison, we set the population size, NP, to 120 with respect to all algorithms. Specifically, for the CCGDE3, the population is split into 2 subpopulations, each of which has 60 individuals. For CMODE, because there are 3 objectives that must be optimized, we use 3 subpopulations, each of which has 20 individuals, and set the maximum size of the archive to 120. For the MOEA/DVA and NSGA-III, we simply set NP to 120. For the DPCCMOLSIA, each of the subpopulations and the archive population has 120 individuals. Finally, we select 120 individuals from all populations.

DE is used in the CCGDE3, and we set $F = 0.5$ and $CR = 1.0$. *SBX* and *polynomial mutation* are used in the MOEA/DVA, NSGA-III and DPCCMOLSIA, and the distribution indexes are set to $\eta_c = \eta_m = 20$. The probabilities of crossover and mutation are set to $p_c = 1.0$ and $p_m = 1.0/D$, respectively.

For MOEA/DVA, the probability of selecting individuals among the neighborhood is 0.9, the neighborhood size is $0.1 \times NP$ and the replace limit is $0.01 \times NP$.

For DVA in MOEA/DVA, the number of control variable analysis is $NCA = 20$ and the number of interdependence analysis is $NIA = 6$. For the variable property and interaction analyses in DPCCMOLSIA, $NCA = 20$ and $NIA = 1$.

Additionally, for the DPCCMOLSIA, we set $N_{sel} = 0.1 \times NP$, and the number of CPUs used is 72, while other algorithms are serial.

4.3. Performance Indicator

Because the optimal solutions are unknown, we use the hypervolume (HV) indicator [47] and visualize all the obtained solution sets. The HV indicator translates the solution set quality into a single evaluation index. The higher the HV indicator value, the better the optimization performance.

4.4. Results and Analyses

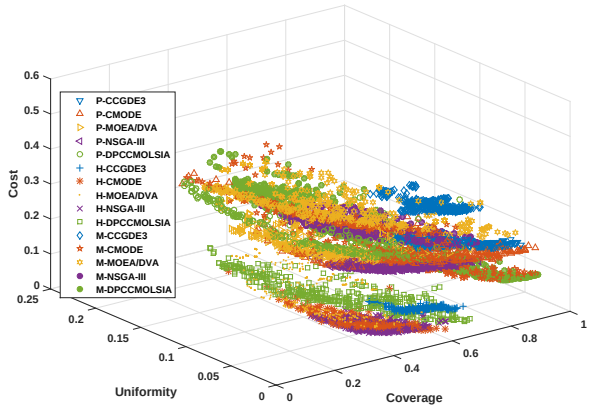


Figure 3: Visualization of solutions for all terrains.

First, we demonstrate all the obtained final nondominated solutions after 24 runs of each algorithm on each of the three terrains in Fig. 3. Here, $P-*$ denotes the results for the plain terrain, $H-*$ denotes the results for the hilly terrain, and $M-*$ denotes the results for the mountainous terrain.

Fig. 3 shows that the characteristics are quite different for the different terrains. In general, for the plain terrain, all the algorithms perform better in terms of *Coverage*. For the hilly

terrain, the algorithms tend to perform well in terms of the *Deployment Cost* objective. Finally, for the mountainous terrain, the performances of the algorithms are far inferior to their performances for the other two terrains. We can comment on the above phenomena as follows:

1. Because the plain terrain is flatter than the other two terrains, it is easier to achieve better *Coverage*.
2. The hilly terrain has fluctuations in elevation, and the algorithms tend to deploy the sensor nodes in low-lying areas, thus guaranteeing better *Deployment Cost*.
3. The mountainous terrain has severe elevation changes, which makes it much more difficult to address compared with the other two terrains. Consequently, the algorithms exhibit poor performances for this terrain.

In the following, we give detailed results of all algorithms with respect to each terrain and provide corresponding performance analyses.

4.4.1. Plain Terrain

The evolutionary curves of the HV indicator values are illustrated in Fig. 4.

We can see that the DPCCMOLSIA has the best performance (0.6864839), followed by the MOEA/DVA (0.6582590), the CMODE (0.6290526), and the NSGA-III (0.5526697); the CCGDE3 has the worst performance (0.3539973). Moreover, the DPCCMOLSIA has the fastest convergence speed, but less improvement occurs in the consequent process, while the MOEA/DVA is quite inferior in the beginning stage but improves significantly in the middle stage.

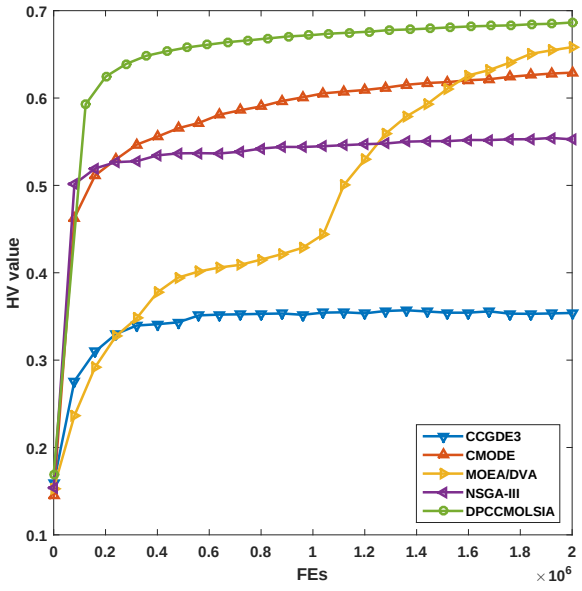
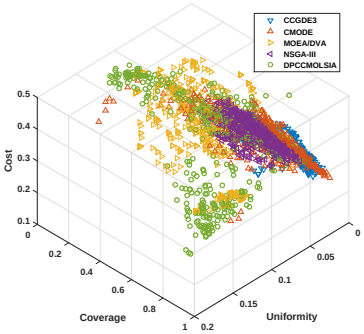
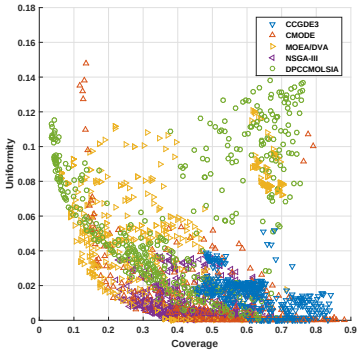


Figure 4: Evolutionary curves of HV indicator values (plain terrain).

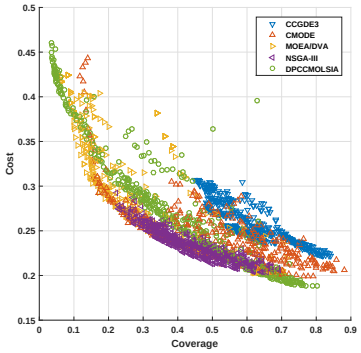
The visualization is shown in Fig. 5. In accordance with the HV indicator and considering the diversity and convergence of



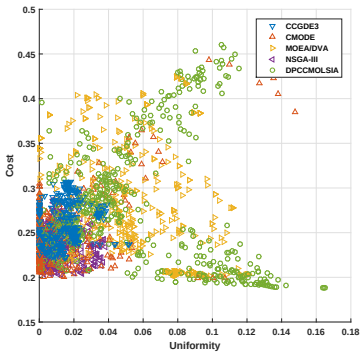
(a)



(b)



(c)



(d)

Figure 5: Visualization of solutions for plain terrain.

solutions, the overall performance of the DPCCMOLSIA is the best.

Coverage is an important factor to consider in WSN deployment problems. From the visualization, we can see that the DPCCMOLSIA is able to obtain a very low fitness value (high coverage rate) for the *Coverage* objective, which validates its performance. Because the plain terrain is quite flat, it is easier to optimize the objectives *Connectivity Uniformity* and *Deployment Cost*.

Overall, the performances of all the algorithms for the plain terrain can be ordered as follows: DPCCMOLSIA > MOEA/DVA > CMODE > NSGA-III > CCGDE3.

4.4.2. Hilly Terrain

The HV indicator value evolutionary curves for all the algorithms for the hilly terrain are illustrated in Fig. 6.

The HV indicator values again reveal that the DPCCMOLSIA has the best performance (0.7894622), followed by the MOEA/DVA (0.7794569), the CMODE (0.7070007), the NSGA-III (0.6374458), and the CCGDE3 (0.4470647). The characteristics of all the algorithms resemble those described above for the plain terrain.

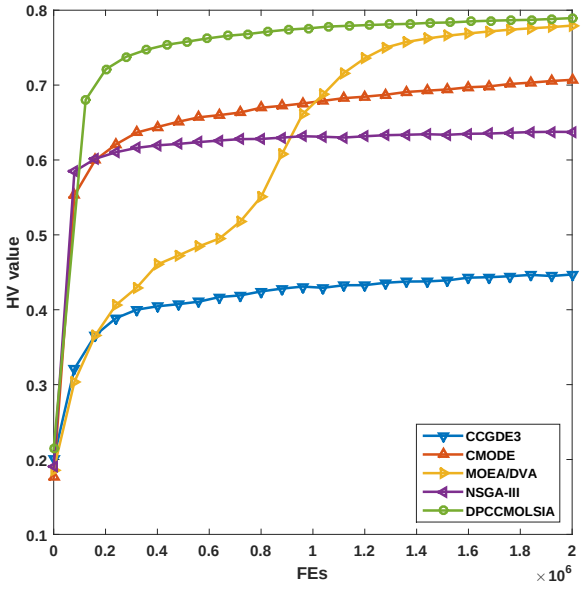


Figure 6: Evolutionary curves of HV indicator values (hilly terrain).

The visualization of the solutions are shown in Fig. 7. Generally, the DPCCMOLSIA more comprehensively approximates the optimal PF and still guarantees good *Coverage*. As mentioned above, because the fluctuations in the hilly terrain are relatively smaller and the flat area is larger compared to the mountainous terrain, the algorithms obtain a relatively good *Deployment Cost*.

Overall, the performances of the algorithms for the hilly terrain can be ordered as follows: DPCCMOLSIA > MOEA/DVA > CMODE > NSGA-III > CCGDE3.

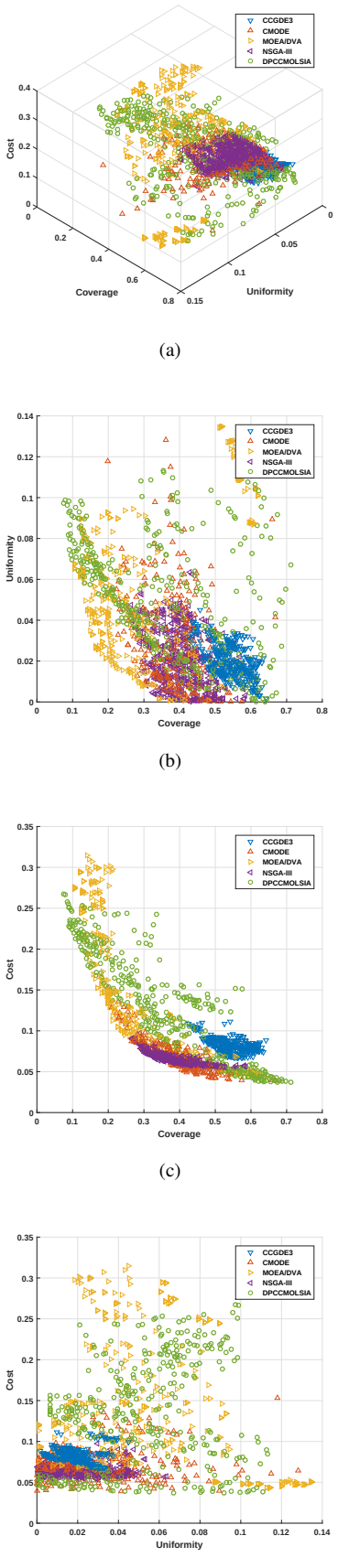


Figure 7: Visualization of solutions for hilly terrain.

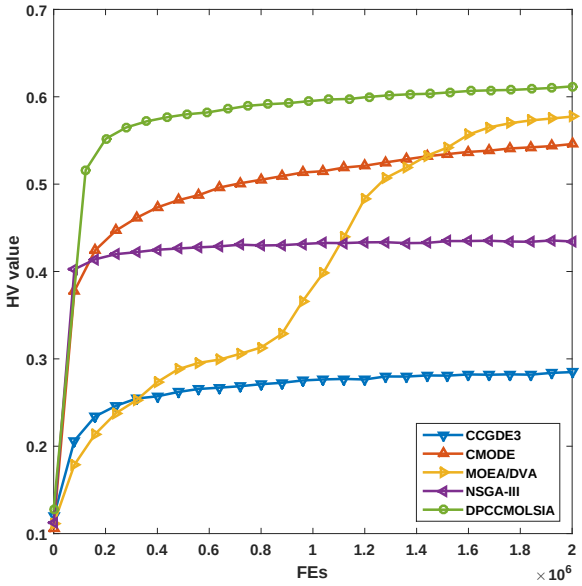


Figure 8: Evolutionary curves of HV indicator values (mountainous terrain).

4.4.3. Mountainous Terrain

The HV indicator value evolutionary curves of the DPCCMOLSIA, the MOEA/DVA, the CMODE, the NSGA-III and the CCGDE3 for the mountainous terrain are illustrated in Fig. 8.

The DPCCMOLSIA again yields the highest HV indicator value (0.6119342), followed by the MOEA/DVA (0.5773018), the CMODE (0.5459146), the NSGA-III (0.4343607), and the CCGDE3 (0.2848895). The characteristics of the different algorithms are similar to those for the plain and hilly terrains.

Visualizations of the nondominated solution sets produced by all the algorithms are illustrated in Fig. 9. Overall, the DPCCMOLSIA performs the best. Because the mountainous terrain has severe altitude variations, it is much more difficult for the algorithms to achieve a good optimization performance.

The performances of all five algorithms for the mountainous terrain can be ordered as follows: DPCCMOLSIA > MOEA/DVA > CMODE > NSGA-III > CCGDE3.

Overall, comprehensively considering all the tested terrains, the DPCCMOLSIA is the best in terms of the optimization results; the MOEA/DVA is inferior; the CMODE is the third; the NSGA-III is fourth; and the CCGDE3 is last.

Table 1 summarizes the computation times required by the various algorithms. Compared to the serial algorithms, the computation time of the DPCCMOLSIA is substantially reduced.

5. Conclusions and Prospects

In the present paper, we put forward a distributed parallel cooperative coevolutionary multi-objective large-scale immune algorithm (DPCCMOLSIA), which uses a three-layer parallel

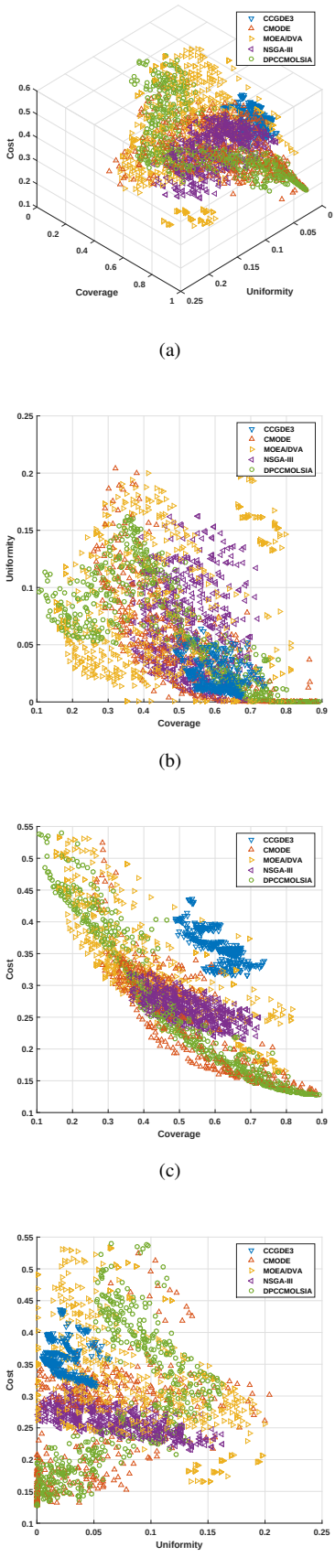


Figure 9: Visualization of solutions for mountainous terrain.

Table 1: Average Computation Time of the CCGDE3, CMODE, MOEA/DVA, NSGA-III and DPCCMOLSIA, and the Speedup Ratios with Respect to the DPCCMOLSIA

AVERAGE TIME	CCGDE3	CMODE	MOEA/DVA	NSGA-III	DPCCMOLSIA
Plain terrain	8.52E+03	8.99E+03	8.67E+03	9.21E+03	**1.64E+02**[1]
Hilly terrain	1.29E+04	1.45E+04	1.14E+04	1.49E+04	**2.37E+02**
Mountainous terrain	9.64E+03	1.31E+04	1.07E+04	1.26E+04	**2.30E+02**
All terrains	3.11E+04	3.66E+04	3.08E+04	3.67E+04	**6.31E+02**
Speedup ratio	4.93E+01	5.80E+01	4.88E+01	5.82E+01	/

[1] Values in bold denote better performance.

structure to substantially reduce the computation time. By decomposing the objectives and variables, the original complex MOLSOP is transformed into simpler, small-scale problems that are easier to address. Via tests on real-world terrain data, compared with several other algorithms (CCGDE3, CMODE, MOEA/DVA and NSGA-III), the DPCCMOLSIA can achieve better optimization results in much less time. In the future, we plan to continue improving the DPCCMOLSIA and to test it on additional real-world problems.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grant No. 61303001, in part by the Foundation of Key Laboratory of Machine Intelligence and Advanced Computing of the Ministry of Education under Grant No. MSC-201602A, in part by the Opening Project of Guangdong High Performance Computing Society under Grant No. 2017060101, and in part by the Special Program for Applied Research on Super Computation of the NSFC-Guangdong Joint Fund (the second phase) under Grant No. U1501501. This work was conducted at the National Supercomputer Center in Guangzhou (NSCC-GZ) and National Supercomputer Center in Tianjin (NSCC-TJ), and the calculations were performed on TianHe-2 and TianHe-1(A). The staff from the supercomputer centers and the engineers from Beijing Paratera Technology Co., Ltd., provided effective support and made the computation process smooth. Thanks for all the support.

References

[1] B. Cao, J. Zhao, Z. Lv, X. Liu, 3D terrain multiobjective deployment optimization of heterogeneous directional sensor networks in security monitoring, Vol. PP, 2017, pp. 1–1. doi:10.1109/TBDATA.2017.2685581.

[2] S. A. Kauffman, Origins of Order in Evolution: Self-Organization and Selection, Springer Netherlands, Dordrecht, 1992, pp. 153–181. doi:10.1007/978-94-015-8054-0_8.
URL http://dx.doi.org/10.1007/978-94-015-8054-0_8

[3] B. Wang, X. Gu, L. Ma, S. Yan, Temperature error correction based on BP neural network in meteorological wireless sensor network, International Journal of Sensor Networks (IJSNET) 23 (4) (2017) 265 – 278. doi:10.1504/IJSNET.2017.083532.

[4] J. Shen, S. Chang, J. Shen, Q. Liu, X. Sun, A lightweight multi-layer authentication protocol for wireless body area networks, Future Generation Computer Systemsdoi:http://dx.doi.org/10.1016/j.future.2016.11.033.
URL http://www.sciencedirect.com/science/article/pii/S0167739X16306963

[5] Y. Zhang, X. Sun, B. Wang, Efficient algorithm for k-barrier coverage based on integer linear programming, China Communications 13 (7) (2016) 16–23. doi:10.1109/CC.2016.7559071.

[6] Z. Zhou, Q. J. Wu, F. Huang, X. Sun, Fast and accurate near-duplicate image elimination for visual sensor networks, International Journal of Distributed Sensor Networks 13 (2) (2017) 1550147717694172. arXiv:http://dx.doi.org/10.1177/1550147717694172, doi:10.1177/1550147717694172.
URL http://dx.doi.org/10.1177/1550147717694172

[7] J. Zhang, J. Tang, T. Wang, F. Chen, Energy-efficient data-gathering rendezvous algorithms with mobile sinks for wireless sensor networks, International Journal of Sensor Networks (IJSNET) 23 (4) (2017) 248 – 257. doi:10.1504/IJSNET.2017.10004216.

[8] L. N. de Castro, F. J. V. Zuben, Learning and optimization using the clonal selection principle, IEEE Transactions on Evolutionary Computation 6 (3) (2002) 239–251. doi:10.1109/TEVC.2002.1011539.

[9] C. A. C. Coello, N. C. Cortés, An approach to solve multiobjective optimization problems based on an artificial immune system, in: International Conference on Artificial Immune Systems, 2002, pp. 212–221.

[10] Y. Xue, J. Jiang, B. Zhao, T. Ma, A self-adaptive artificial bee colony algorithm based on global best for global optimization, Soft Computing (8) (2017) 1–18. doi:10.1007/s00500-017-2547-1.
URL https://doi.org/10.1007/s00500-017-2547-1

[11] E. Zitzler, M. Laumanns, L. Thiele, SPEA2: Improving the strength Pareto evolutionary algorithm, Tech. rep., Eidgenössische Technische Hochschule Zürich (ETH), Institut für Technische Informatik und Kommunikationsnetze (TIK) (2001). doi:10.3929/ethz-a-004284029.

[12] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2) (2002) 182–197. doi:10.1109/4235.996017.

[13] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Transactions on Evolutionary Computation 11 (6) (2007) 712–731. doi:10.1109/TEVC.2007.892759.

[14] T. Zhu, W. Luo, C. Bu, L. Yue, Accelerate population-based stochastic search algorithms with memory for optima tracking on dynamic power systems, IEEE Transactions on Power Systems 31 (1) (2016) 268–277. doi:10.1109/TPWRS.2015.2407899.

[15] C. Bu, W. Luo, L. Yue, Continuous dynamic constrained optimization with ensemble of locating and tracking feasible regions strategies, IEEE Transactions on Evolutionary Computation 21 (1) (2017) 14–33. doi:10.1109/TEVC.2016.2567644.

[16] C. Bu, W. Luo, T. Zhu, L. Yue, Solving online dynamic time-linkage problems under unreliable prediction, Applied Soft Computing 56 (2017) 702 – 716. doi:http://dx.doi.org/10.1016/j.asoc.2016.11.005.
URL http://www.sciencedirect.com/science/article/pii/S1568494616305749

[17] Y. Zhang, W. Luo, Z. Zhang, B. Li, X. Wang, A hardware/software partitioning algorithm based on artificial immune principles, Applied Soft Computing 8 (1) (2008) 383 – 391. doi:http://dx.doi.org/10.1016/j.asoc.2007.03.003.
URL http://www.sciencedirect.com/science/article/pii/S1568494607000257

[18] W. Luo, J. Sun, C. Bu, H. Liang, Species-based particle swarm optimizer enhanced by memory for dynamic optimization, Applied Soft Computing 47 (2016) 130 – 140. doi:http://dx.doi.org/10.1016/j.asoc.2016.05.032.
URL http://www.sciencedirect.com/science/article/pii/S1568494616302423

[19] J. Yoo, P. Hajela, Immune network simulations in multicriterion design, Structural optimization 18 (2) (1999) 85–94. doi:10.1007/BF01195983.
URL http://dx.doi.org/10.1007/BF01195983

[20] M. Gong, L. Jiao, H. Du, L. Bo, Multiobjective immune algorithm with nondominated neighbor-based selection, Evolutionary Computation 16 (2) (2008) 225–255. doi:10.1162/evco.2008.16.2.225.

[21] Q. Lin, J. Chen, Z. H. Zhan, W. N. Chen, C. A. C. Coello, Y. Yin, C. M. Lin, J. Zhang, A hybrid evolutionary immune algorithm for multiobjec-

tive optimization problems, IEEE Transactions on Evolutionary Computation 20 (5) (2016) 711–729. doi:10.1109/TEVC.2015.2512930.

[22] Y.-J. Gong, W.-N. Chen, Z.-H. Zhan, J. Zhang, Y. Li, Q. Zhang, J.-J. Li, Distributed evolutionary algorithms and their models, Appl. Soft Comput. 34 (C) (2015) 286–300. doi:10.1016/j.asoc.2015.04.061.
URL http://dx.doi.org/10.1016/j.asoc.2015.04.061

[23] B. Cao, J. Zhao, Z. Lv, X. Liu, A distributed parallel cooperative coevolutionary multiobjective evolutionary algorithm for large-scale optimization, IEEE Transactions on Industrial Informatics 13 (4) (2017) 2030–2038. doi:10.1109/TII.2017.2676000.

[24] Z. H. Zhan, X. F. Liu, H. Zhang, Z. Yu, J. Weng, Y. Li, T. Gu, J. Zhang, Cloudde: A heterogeneous differential evolution algorithm and its distributed cloud version, IEEE Transactions on Parallel and Distributed Systems 28 (3) (2017) 704–716. doi:10.1109/TPDS.2016.2597826.

[25] D. A. V. Veldhuizen, J. B. Zydallis, G. B. Lamont, Considerations in engineering parallel multiobjective evolutionary algorithms, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 144–173. doi:10.1109/TEVC.2003.810751.

[26] A. J. Nebro, J. J. Durillo, A Study of the Parallelization of the Multi-Objective Metaheuristic MOEA/D, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 303–317. doi:10.1007/978-3-642-13800-3_32.
URL http://dx.doi.org/10.1007/978-3-642-13800-3_32

[27] J. J. Durillo, Q. Zhang, A. J. Nebro, E. Alba, Distribution of Computational Effort in Parallel MOEA/D, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 488–502. doi:10.1007/978-3-642-25566-3_38.
URL http://dx.doi.org/10.1007/978-3-642-25566-3_38

[28] J. Ge, Z. Chen, Y. Wu, Y. E, H-SOFT: a heuristic storage space optimisation algorithm for flow table of openflow, Concurrency and Computation: Practice and Experience 27 (13) (2015) 3497–3509, cPE-13-0288.R1. doi:10.1002/cpe.3206.
URL http://dx.doi.org/10.1002/cpe.3206

[29] B. Cao, J. Zhao, Z. Lv, X. Liu, S. Yang, X. Kang, K. Kang, Distributed parallel particle swarm optimization for multi-objective and many-objective large-scale optimization, IEEE Access 5 (2017) 8214–8221. doi:10.1109/ACCESS.2017.2702561.

[30] X. Ma, F. Liu, Y. Qi, X. Wang, L. Li, L. Jiao, M. Yin, M. Gong, A multiobjective evolutionary algorithm based on decision variable analyses for multiobjective optimization problems with large-scale variables, IEEE Transactions on Evolutionary Computation 20 (2) (2016) 275–298. doi:10.1109/TEVC.2015.2455812.

[31] J. J. Escobar, J. Ortega, J. González, M. Damas, Assessing Parallel Heterogeneous Computer Architectures for Multiobjective Feature Selection on EEG Classification, Springer International Publishing, Cham, 2016, pp. 277–289. doi:10.1007/978-3-319-31744-1_25.
URL http://dx.doi.org/10.1007/978-3-319-31744-1_25

[32] A. G. D. Nuovo, M. Palesi, V. Catania, Multi-objective evolutionary fuzzy clustering for high-dimensional problems, in: 2007 IEEE International Fuzzy Systems Conference, 2007, pp. 1–6. doi:10.1109/FUZZY.2007.4295660.

[33] Y. Zuo, M. Gong, J. Zeng, L. Ma, L. Jiao, Personalized recommendation based on evolutionary multi-objective optimization [research frontier], IEEE Computational Intelligence Magazine 10 (1) (2015) 52–62. doi:10.1109/MCI.2014.2369894.

[34] M. A. Potter, K. A. De Jong, A cooperative coevolutionary approach to function optimization, Springer Berlin Heidelberg, Berlin, Heidelberg, 1994, pp. 249–257. doi:10.1007/3-540-58484-6_269.
URL http://dx.doi.org/10.1007/3-540-58484-6_269

[35] F. van den Bergh, A. P. Engelbrecht, A cooperative approach to particle swarm optimization 8 (3) (2004) 225–239. doi:10.1109/TEVC.2004.826069.

[36] M. N. Omidvar, X. Li, X. Yao, Cooperative co-evolution with delta grouping for large-scale non-separable function optimization, in: Proc. IEEE Congr. Evol. Comput., 2010, pp. 1–8. doi:10.1109/CEC.2010.5585979.

[37] X. Li, X. Yao, Cooperatively coevolving particle swarms for large-scale optimization, IEEE Trans. Evol. Comput. 16 (2) (2012) 210–224. doi:10.1109/TEVC.2011.2112662.

[38] M. N. Omidvar, X. Li, Y. Mei, X. Yao, Cooperative co-evolution with differential grouping for large-scale optimization, IEEE Trans. Evol. Comput. 18 (3) (2014) 378–393. doi:10.1109/TEVC.2013.2281543.

[39] Y. Mei, M. N. Omidvar, X. Li, X. Yao, A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization, ACM Trans. Math. Softw. 42 (2) (2016) 13:1–13:24. doi:10.1145/2791291.
URL http://doi.acm.org/10.1145/2791291

[40] Y. Ling, H. Li, B. Cao, Cooperative co-evolution with graph-based differential grouping for large scale global optimization, in: 2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2016, pp. 95–102. doi:10.1109/FSKD.2016.7603157.

[41] L. M. Antonio, C. A. C. Coello, Use of cooperative coevolution for solving large scale multiobjective optimization problems, in: 2013 IEEE Congress on Evolutionary Computation, 2013, pp. 2758–2765. doi:10.1109/CEC.2013.6557903.

[42] S. Huband, P. Hingston, L. Barone, L. While, A review of multiobjective test problems and a scalable test problem toolkit, IEEE Trans. Evol. Comput. 10 (5) (2006) 477–506. doi:10.1109/TEVC.2005.861417.

[43] J. Wang, W. Zhang, J. Zhang, Cooperative differential evolution with multiple populations for multiobjective optimization, IEEE Transactions on Cybernetics 46 (12) (2016) 2848–2861. doi:10.1109/TCYB.2015.2490669.

[44] Y. Wu, G. Min, K. Li, B. Javadi, Modeling and analysis of communication networks in multicluster systems under spatio-temporal bursty traffic, IEEE Transactions on Parallel and Distributed Systems 23 (5) (2012) 902–912. doi:10.1109/TPDS.2011.198.

[45] Y. Wu, G. Min, D. Zhu, L. T. Yang, An analytical model for on-chip interconnects in multimedia embedded systems, ACM Trans. Embed. Comput. Syst. 13 (1s) (2013) 29:1–29:19. doi:10.1145/2536747.2536751.
URL http://doi.acm.org/10.1145/2536747.2536751

[46] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints, IEEE Transactions on Evolutionary Computation 18 (4) (2014) 577–601. doi:10.1109/TEVC.2013.2281535.

[47] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, V. G. da Fonseca, Performance assessment of multiobjective optimizers: An analysis and review, IEEE Transactions on Evolutionary Computation 7 (2) (2003) 117–132. doi:10.1109/TEVC.2003.810758.