Decentralized Estimation of Distribution Algorithms for Large Scale Networked Systems*

Aizhe Bian
Center for Intelligent and Networked Systems
Department of Automation, Tsinghua University
Beijing, China
baz16@mails.tsinghua.edu.cn

Xi Chen
Center for Intelligent and Networked Systems
Department of Automation, Tsinghua University
Beijing, China
bjchenxi@tsinghua.edu.cn

***Abstract*—In order to solve optimization problems in large scale networked systems, this paper proposes a method to implement estimation of distribution algorithms (EDA) in a decentralized way. The main point of decentralized EDA is that each subsystem solves its own optimization problems based on local and its neighbors' information. Numerical examples illustrate the effectiveness of the algorithm.**

***Index Terms*—Large scale networked systems, decentralized optimization, estimation of distribution algorithms**

I. Introduction

Nowadays, large scale networked systems like electrical power grids, the intelligent building systems, the heating, ventilation, and conditioning (HVAC) systems, and unmanned aerial vehicle (UAV) formation systems are playing important roles in our life. Meanwhile, the scale of these systems is expanding and the size is extending. The expansion of scale results in the great increase of computational complexity. Centralized optimization algorithms, which require for information from the whole system, have become increasing helpless on this condition. Moreover, centralized algorithms are sensitive to the change of systems' topologies. Therefore, decentralized optimization algorithms are worthwhile to be researched [1].

A large scale networked system is combined by multiple subsystems. These subsystems may be physically dispersed. Each subsystem has a local optimization objective and a series of local decision variables. The subsystems coupled with their neighbors by sharing parts of decision variables. The global decision vector is the combination of local decision variables. The objective is to minimize the summation of objectives of all subsystems. For example, in an intelligent building system, each room can be viewed as a subsystem. The objective is to minimize the energy consumption of the building system.

For decentralized algorithms, each subsystem uses the same algorithm to solve its own optimization problem based on its local and neighbors' information. Because subsystems don't have all information of the system, the decentralized algorithms need to use communication methods to find the optimal solutions for the whole system [2].

Decentralized methods have been proposed since the 1980s [3]. Bertsekas and Tsitsiklis present many iterative algorithms to solve decentralized problems and proposed the conditions of convergence of these methods [3]. Inspired by the rule of "local autonomy and communication with neighbors", many researchers proposed synchronous decentralized algorithms [2]. In 2002, Inalhan et al. present a decentralized algorithm to optimize a problem and prove its convergence under some assumptions [4]. In 2009, Yang et al. proposed a decentralized optimization algorithm to optimize the multiagent system-based watershed management [5]. In 2010, Loureiro et al. present an approach for managing shared resource pools in a decentralized, adaptive and optimal way [6]. In 2015, Hu and Chen proposed a decentralized ordinal optimization for optimization problems in networked system and apply adaptive learning in the method to narrow down the search space [7].

Apart from synchronous algorithms, in 2008, Huang et al. present an asynchronous decentralized algorithm for interconnected electricity markets [8].

Estimation of distribution algorithms (EDAs) are a series of evolution algorithms based on statistics [9]–[11]. The main idea of EDAs is to estimate the distribution of good solutions selected from the generation and then generate the next generation by sampling based on the distribution. EDAs are population based algorithms that discard some individuals each generation and replace them using the statistical properties of highly-fit individuals [9].

This paper focus on the decentralized EDA to solve optimization problems in large scale networked systems. We present a new decentralized algorithm based on EDAs. Networked system can be divided into several subsystems according to its physical property. The algorithm makes each subsystem solve its own optimization problem independently based on local and neighbors' information. That is to say, all subsystems solve their problems in parallel and make the whole system in optimization state by communication methods. Therefore, it can solve more complex problems in a shorter time than an EDA.

The rest of this paper is organized as follows. We formulate the problem in Section 2. Then we present our algorithms in Section 3. In Section 4, we use some benchmark problems to show the performance of the algorithm compared with centralized algorithms. In the end, we conclude our work in Section 5.

This work was partially supported by the National Key Research and Development Project of China (No.20171231799).

978-1-7281-1312-8/18/$31.00 ©2018 IEEE

II. Problem Formulation

A large scale networked system contains many subsystems and these subsystems may connect with each other. Therefore, the whole system can be formulated as a graph $G(V, E)$, where V is the set of all subsystems (vertexes) and E is the set of all connections (edges) between subsystems. An optimization problem usually can be described as a minimization problem. For the large scale networked system, the goal of the minimization problem is to minimize the sum of the loss functions of all subsystems.

Each subsystem has its own decision variables and loss function and only the subsystem itself can decide the value of its own decision variables. However, decision variables of neighbors are involved when calculating the loss function. That is, the subsystem needs to exchange information with its neighbors. The decentralized approach let subsystems solve their own problem in parallel without any central coordination.

A. Notations

N: population size of each subsystem;
M: number of selected individuals to estimate distribution;
$G(V, E)$: graph of the system;
$B(i)$: the set of neighbors of subsystem i,
$B(i) = \{j \in V : (i, j) \in E\}$;
$\widetilde{B}(i)$: $\widetilde{B}(i) = B(i) \cup \{i\}$;
$\mathbf{x}$: decision variables of system
$\mathbf{x}_i$: decision variables of subsystem i,
$\mathbf{x}_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,D_i})^T$;
$\mathbf{x}_i \oplus \mathbf{x}_j$: $(x_{i,1}, x_{i,2}, \ldots, x_{i,D_i}, x_{j,1}, x_{j,2}, \ldots, x_{j,D_j})^T$;
$\widetilde{\mathbf{x}}_i$: $\oplus_{j \in \widetilde{B}(j)} \mathbf{x}_j$, subsystem and its neighbors variables;

B. Mathematical Model

The unconstrained optimization problem for a large scale networked system containing n subsystems can be described as follows:

$$\begin{aligned} \min_{\mathbf{x}} F(\mathbf{x}) &= \sum_{\mathbf{x}} F(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \sum_{i=1}^{n} f_i(\widetilde{\mathbf{x}}_i) \\ &= f_1(\widetilde{\mathbf{x}}_1) + f_2(\widetilde{\mathbf{x}}_2) + \cdots + f_n(\widetilde{\mathbf{x}}_n) \end{aligned} \tag{1}$$

where vector $\mathbf{x}_i$ means the variable belongs to subsystem i and f_i means the loss function of it. The goal is to minimize F(the sum of the loss function of each subsystem). However, some subsystems have connections to each other, which means they share decision variables with their neighbors. In order to describe this situation, we use variable completion method.

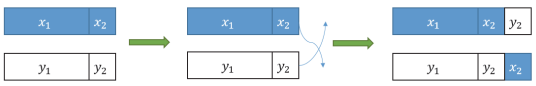

Fig. 1. Variable completion method

Figure 1 shows the variable completion method. For these two subsystems, their own decision variables are $[x_1, x_2]$ and $[y_1, y_2]$. However, the decision variables of optimization problems are $\widetilde{\mathbf{x}} = [x_1, x_2, y_2]$ and $\widetilde{\mathbf{y}} = [y_1, y_2, x_2]$. They exchange part of decision variables with each other. As is shown in equation (1), we use $\widetilde{\mathbf{x}}_i$ to describe the variables needed in subsystem i. $\widetilde{\mathbf{x}}_i$ contains variables in subsystem i and its neighbors.

For decentralized optimization, the core point is to make subsystems optimize their own problems separately. For each subsystem, the optimization problem is: $\min_{\mathbf{x}_i} f_i(\widetilde{\mathbf{x}}_i)$.

III. Decentralized Estimation of Distribution Algorithms

A. Estimation of Distribution Algorithms

Estimation of Distribution Algorithms(EDAs) are a series of algorithms used to optimize a function by keeping track of statistics of the population of candidate solutions [9].

An EDA is a kind of evolution algorithms. However, in an EDA there are neither crossover nor mutation operators. Instead, the new population of individuals is sampled from a probability distribution, which is estimated from a database containing selected individuals from the previous generation. The process repeats from one generation to the next until we get the proper solutions. The framework of an EDA is shown as follows.

Algorithm 1 Framework of an EDA

1: Initialize a population of candidate solutions, population size is N
2: **while** not termination criterion **do**
3: Compute the loss function value of each individual;
4: Select M individuals from the population based on some rules;
5: Estimate the distribution of good solutions based on the M individuals;
6: Sample from the distribution to generate the new generation;
7: **end while**
8: Choose the solution with least loss value.

For some real-world problems, the loss is measured rather than calculated. What we get is a set of data and we can't use the gradient method to solve the optimization problem. An EDA doesn't need the gradient information. It can be more widely applied.

EDAs have different ways for estimating the distribution. Here we use the Univariate Marginal Distribution Algorithm(UMDA). It uses binary coding and Bernoulli distribution to estimate the whole distribution [9].

B. Decentralized EDA

We present a decentralized EDA based on EDAs. The algorithm flow is shown as follows.

We change the structure of EDAs so that the decentralized EDA can be suitable for large scale networked systems. As is mentioned before, the subsystem need neighbors' decision variables to calculate its loss function. So we use distribution of neighbors to generate N individuals in step 5. Then these

Algorithm 2 Framework of the decentralized EDA

1: **for** subsystem i **do**
2: Initialize a population of candidate solutions for each subsystem, population size is N;
3: **while** not termination criterion **do**
4: Exchange distributions with neighbors;
5: Generate complete variables for each individuals based on neighbors' distribution;
6: Calculate the loss value of each individual;
7: Select $2M$ individuals with least loss values($M \ll N$);
8: For each selected individuals, calculate the influence on the whole system;
9: Calculate the cost by loss and influence;
10: Select M individuals based on cost from the $2M$ individuals;
11: Calculate the distribution based on these M individuals;
12: Generate the local individuals;
13: **end while**
14: **end for**
15: Get the solutions

new individuals can combine with local individuals so that we get N complete decision variables.

Subsystems need to cooperate with each other to adjust the value of local decision variable to minimize the loss function. So we use cost instead of loss to select the good individuals in step 10.

$$cost = loss + influence \tag{2}$$

Then we need to calculate the influence in equation (3). We can change the form of the whole loss function as follows:

$$\begin{aligned}\min_{\mathbf{x}_i} f(\mathbf{x}) &= \sum_{j\in\widetilde{B}(i)} f_j(\widetilde{\mathbf{x}}_j) + \sum_{j\notin\widetilde{B}(i)} f_j(\widetilde{\mathbf{x}}_j) \\ &= f_i(\widetilde{\mathbf{x}}_i) + \sum_{j\in B(i)} f_j(\widetilde{\mathbf{x}}_j) + \sum_{j\notin\widetilde{B}(i)} f_j(\widetilde{\mathbf{x}}_j)\end{aligned} \tag{3}$$

The second formula $\sum_{j\in B(i)} f_j(\widetilde{\mathbf{x}}_j)$ represents the influence on neighbors. But we can't calculate it directly. For each individual in subsystem i, we can find part of decision variables x_{ji} that belongs to subsystem j. Then we can find k individuals closest to x_{ji} in subsystem j. We use $\mathbf{x}_{\hat{j}}$ to represent these k individuals. Then we can use the average loss values of these individuals to represent the influence as is shown in (5).

$$influence(\mathbf{x}_i) = \sum_{j\in B(i)} (\sum_{k} f_j(\mathbf{x}_{\hat{j}})/k) \tag{4}$$

We will discuss the computational complexity of the decentralized EDA. We ignore the cost of data exchange between neighbors because it depends on the communication network. For each subsystem, we need to calculate N loss function values in an iteration. Compared with centralized EDA, we needn't calculate extra loss function values in an iteration. However, we need find some individuals and calculate the average solution of them in an iteration. These operations need much less computational resource than calculating loss functions. Meanwhile, the information transfer between neighbors also need extra operations. So the computational complexity of a decentralized EDA and a centralized EDA is similar in an iteration if they have the same parameters. Considering that the computing resource is distributed on each subsystem, the decentralized EDA can perform better.

IV. Experimental Results

We apply our method in systems with different topologies to verify the performance of our method. We consider two topologies: chain and grid. For these subsystems, we use different benchmarks as their optimization objectives.

A. Example 1: topology of chain

The topology is shown in Figure 2.

Fig. 2. 4 subsystems with a topology of chain

The loss functions are as follows:
Subsystem 1: the Rastrigin function;

$$\begin{aligned}\min f_1(\mathbf{x}) =& 10\times 4 + x_{11}^2 + x_{12}^2 + x_{13}^2 + x_{23}^2 \\ &- 10\times(\cos(2\pi x_{11}) + \cos(2\pi x_{12}) \\ &+ \cos(2\pi x_{13}) + \cos(2\pi x_{23}))\end{aligned} \tag{5}$$

Subsystem 2: the quartic function;

$$\min f_2(\mathbf{x}) = x_{21}^4 + 2x_{22}^4 + 3x_{23}^4 + 4x_{13}^4 + 5x_{33}^4 \tag{6}$$

Subsystem 3: the Griewank function;

$$\begin{aligned}\min f_3(\mathbf{x}) =& 1 + (x_{31}^2 + x_{32}^2 + x_{33}^2 + x_{23}^2 + x_{43}^2)/4000 \\ &- [\cos(x_{31})\cos(x_{32}/\sqrt{2})\cos(x_{33}/\sqrt{3}) \\ &\cos(x_{23}/\sqrt{4})\cos(x_{43}/\sqrt{5})]\end{aligned} \tag{7}$$

Subsystem 4: the Schwefel absolute function;

$$\begin{aligned}\min f_4(\mathbf{x}) =& |x_{41}| + |x_{42}| + |x_{43}| + |x_{33}| \\ &+ |x_{41}x_{42}x_{43}x_{33}|\end{aligned} \tag{8}$$

The loss functions of subsystem 1, subsystem 2 and subsystem 4 are multimodal. The loss function f_4 is nonderivable. The overall loss of this problem $F(\mathbf{x}) = \sum_{i=1}^{4} f_i(\mathbf{x})$ is a multimodal function and nonderivable. For the system, the optimal solution is $\mathbf{x}^* = \mathbf{0}, F(\mathbf{x}^*) = 0$.

The overall loss function $F(\mathbf{x})$ is a 12-dimensional function. We use ten bits per dimension, so the loss function includes 120 bits. We use a domain of $[-5, 5]$ for each dimension, the resolution is $10/(2^{10}-1) = 0.0098$. Since the loss function includes 120 bits, we can set population size $N = 1000$ as

a rule of thumb. And we can set elitism parameter $E = 2$, which means that we always keep the best two individuals from one generation to the next. The number of selected individuals to estimate the distribution is $M = 0.1N$ according to the experience. For decentralized EDA, the number of first selection in step 7 is $2M$. We set $k = N/(2M)$.

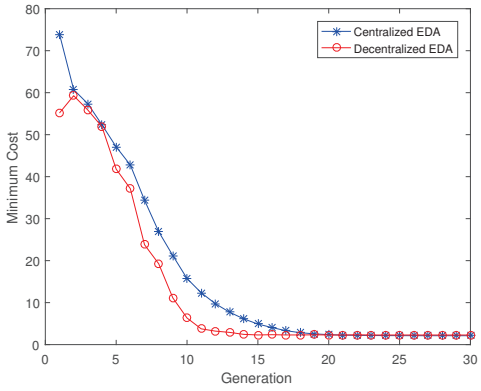

Fig. 3. Example 1: Results for the system

Figure 3 shows the best individual at each generation, averaged over 40 Monte Carlo simulations. We can find that the decentralized EDA converges faster than the centralized EDA. More details of the solutions and decision variables are shown in the following figure and tables.

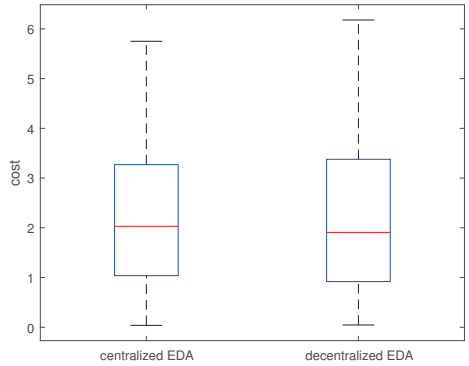

Fig. 4. Boxplot of Centralized EDA and Decentralized EDA

Figure 4 is the boxplot of the 40 Monte Carlo simulation results for these two algorithms. Each box shows the middle 50% of the set of results for an algorithm. The line inside each box shows the median result. The lines above and below each box show the maximum and minimum results.

From Figure 4 and Table I we can find that the two algorithms perform similarly. The best solution and average solution of the centralized EDA is better than the decentralized EDA. The median solution of the decentralized EDA is better. The standard deviation shows that the centralized EDA is more robust than the decentralized EDA. The condition of best decision variables in 40 simulations is shown in the next table.

TABLE I
COMPARISON BETWEEN CENTRALIZED EDA AND DECENTRALIZED EDA

Algorithm	Best solution	Median solution	Average solution	Standard deviation
centralized EDA	0.0385	2.0304	2.1152	1.4830
decentralized EDA	0.0461	1.9076	2.2854	1.8755
relative errors	19.74%	-6.05%	8.05%	30.42%

TABLE II
COMPARISON OF DISTANCE TO OPTIMAL DECISION VARIABLES BETWEEN CENTRALIZED EDA AND DECENTRALIZED EDA

Algorithm	Least distance	Median distance	Average distance	Standard deviation
centralized EDA	0.0049	1.5969	1.5337	0.3394
decentralized EDA	1.5116	1.5909	1.5915	0.0266

The optimal decision variable is $\mathbf{0}$. We calculate the Euclidean distance between our decision variables and the optimum in each simulation. Because the resolution is $10/(2^{10} - 1) = 0.0098$, we can't get optimal decision variable $\mathbf{0}$. From Table II we can find the average distance and median distance is similar.

The number of loss function evaluations of the decentralized EDA is the same as the centralized method in an iteration. As the decentralized EDA makes all subsystems optimize their local problems in parallel, the decentralized EDA converges faster than the centralized EDA.

B. Example 2: topology of grid

The topology is shown in Figure 5.

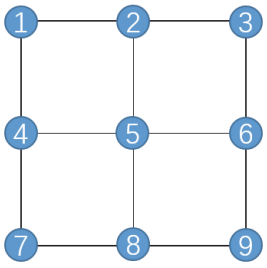

Fig. 5. 9 subsystems with a topology of grid

This topology is more complex. The dimension of decision variables in subsystem 1,3,5,7 and 9 is two. The dimension in subsystem 2,4,6 and 8 is one. In this problem, we use 7 kinds of benchmark functions as follows:
Subsystem 1: the Ackley function;

$$\begin{aligned}\min f_1(\mathbf{x}) =& 20 + e - 20\exp[-0.2(x_{11}^2 + x_{12}^2 + x_2^2 + x_4^2)/4] \\ & - \exp\{[\cos(2\pi x_{11}) + \cos(2\pi x_{12}) \\ & + \cos(2\pi x_2) + \cos(2\pi x_4)]/4\}\end{aligned} \tag{9}$$

Subsystem 2: the tenth power function;

$$\min f_2(\mathbf{x}) = x_2^{10} + x_{12}^{10} + x_{32}^{10} + x_{52}^{10} \tag{10}$$

Subsystem 3: the Griewank function;

$$\begin{aligned}\min f_3(\mathbf{x}) =& 1 + (x_{31}^2 + x_{32}^2 + x_2^2 + x_6^2)/4000 \\ & - [\cos(x_{31})\cos(x_{32}/\sqrt{(2)}) \\ & \cos(x_2/\sqrt{3})\cos(x_6/\sqrt{4})]\end{aligned} \tag{11}$$

Subsystem 4: the Ackley function;

$$\begin{aligned}\min f_4(\mathbf{x}) =& 20 + e - 20\exp[-0.2(x_4^2 + x_{12}^2 + x_{52}^2 + x_{72}^2)/4] \\ & - \exp\{[\cos(2\pi x_4) + \cos(2\pi x_{12}) \\ & + \cos(2\pi x_{52}) + \cos(2\pi x_{72})]/4\}\end{aligned} \tag{12}$$

Subsystem 5: the sphere function;

$$\min f_5(\mathbf{x}) = x_{51}^2 + x_{52}^2 + x_2^2 + x_4^2 + x_6^2 + x_8^2 \tag{13}$$

Subsystem 6: the Rastrigin function;

$$\begin{aligned}\min f_6(\mathbf{x}) =& 10 \times 4 + x_6^2 + x_{32}^2 + x_{52}^2 + x_{92}^2 \\ & - 10 \times (\cos(2\pi x_5) + \cos(2\pi x_{32}) \\ & + \cos(2\pi x_{52}) + \cos(2\pi x_{92}))\end{aligned} \tag{14}$$

Subsystem 7: the quartic function;

$$\min f_7(\mathbf{x}) = x_{71}^4 + 2x_{72}^4 + 3x_4^4 + 4x_8^4 \tag{15}$$

Subsystem 8: the Griewank function;

$$\begin{aligned}\min f_8(\mathbf{x}) =& 1 + (x_8^2 + x_{52}^2 + x_{72}^2 + x_{92}^2)/4000 \\ & - [\cos(x_8)\cos(x_{52}/\sqrt{2}) \\ & \cos(x_{72}/\sqrt{3})\cos(x_{92}/\sqrt{4})]\end{aligned} \tag{16}$$

Subsystem 9: the Schwefel absolute function;

$$\begin{aligned}\min f_9(\mathbf{x}) =& |x_{91}| + |x_{92}| + |x_6| + |x_8| \\ & + |x_{91}x_{92}x_6x_8|\end{aligned} \tag{17}$$

The overall loss of this problem $F(\mathbf{x}) = \sum_{i=1}^{9} f_i(\mathbf{x})$ is a 14-dimensional function. This function is a multimodal function and nonderivable. We use ten bits per dimension, so the loss function includes 140 bits. We use a domain of $[-5, 5]$ for each dimension and the resolution is 0.0098. For the system, the optimal solution is $\mathbf{x}^* = 0, F(\mathbf{x}^*) = 0$. We set $N = 1000, E = 2, M = 0.1N$ and $k = N/(2M)$ like the former problem. The results are as follows.

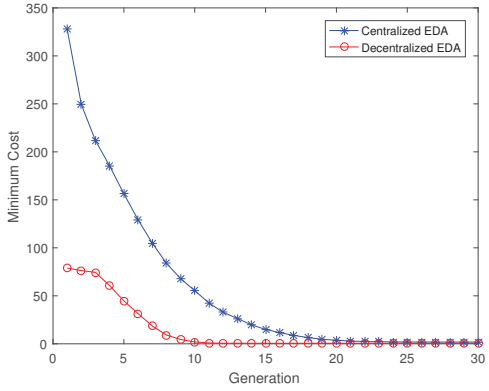

Fig. 6. Example 2: Results for the system

Figure 6 shows the best individual at each generation, averaged over 40 Monte Carlo simulations. From Figure 6, we can find that the decentralized EDA converges faster than the centralized EDA. More details of the solution and decision variables are shown in the following figure and tables.

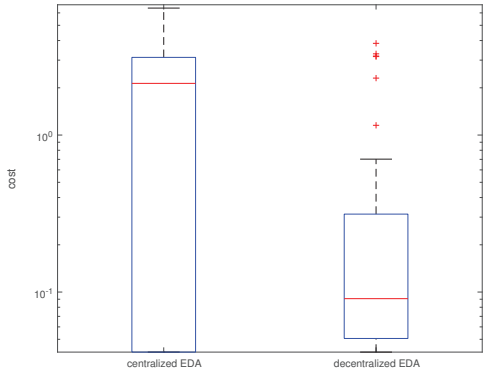

Fig. 7. Boxplot of 40 Monte Carlo Simulation Results for Centralized EDA and Decentralized EDA

TABLE III
COMPARISON BETWEEN CENTRALIZED EDA AND DECENTRALIZED EDA

Algorithm	Best solution	Median solution	Average solution	Standard deviation
centralized EDA	0.0415	2.1336	1.8054	1.8932
decentralized EDA	0.0415	0.0908	0.05483	1.0369
relative errors	0	-95.74%	-69.63%	-45.23%

From Figure 7 and Table IV we can find that the decentralized EDA can find better solutions than the centralized EDA in this problem. The median solution, average solution and standard deviation of the decentralized EDA are smaller than the centralized EDA. The condition of decision variables is shown in the next table.

TABLE IV
COMPARISON OF DISTANCE TO OPTIMAL DECISION VARIABLES BETWEEN CENTRALIZED EDA AND DECENTRALIZED EDA

Algorithm	Least distance	Median distance	Average distance	Standard deviation
centralized EDA	0.0183	0.9337	0.7010	0.8535
decentralized EDA	0.0049	0.0072	0.0426	0.0816

From Table IV we can find that decision variables of the decentralized EDA is closer to the optimal decision variables $\mathbf{0}$.

Compared to the previous example, we can find the performance of the decentralized EDA is better here. We assume that it is related to searching ability. In this problem, there are 9 subsystems and the loss function is complex. Each subsystem has N independent candidate solutions. All subsystems share the distribution and some candidate solutions with neighbors. That means the whole system can search more candidate solutions in an iteration in the decentralized EDA. The searching ability increases with the expansion of scale. In addition, the information from neighbors make the candidate solutions away from local optimum. Subsystems in this example have at least 2 neighbors and subsystems in the previous example have at most 2 neighbors. More neighbors mean more information.

These two examples show that the performance of the decentralized EDA is close to the centralized EDA. In addition, the decentralized EDA converges faster than the centralized EDA.

V. CONCLUSIONS

In this paper, decentralized EDAs is developed to solve optimization problems for large scale networked systems. Compared to centralized optimization algorithm, the decentralized algorithm can solve more complex problems in a shorter time. Besides, decentralized method is more robust when systems change. Thus, decentralized EDA can be widely applied to complex problems in large scale networked systems.

With penalty function, we can use this decentralized EDA to solve optimization problems with constraints. This will be our future work.

REFERENCES

[1] S. Wang, J. Xing, Z. Jiang, and J. Li, "Decentralized optimization for a novel control structure of hvac system," *Mathematical Problems in Engineering*, vol. 2016, 2016.

[2] F. Bullo, J. Cortés, and S. Martinez, *Distributed control of robotic networks: a mathematical approach to motion coordination algorithms*. Princeton University Press, 2009.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.

[4] G. Inalhan, D. M. Stipanovic, and C. J. Tomlin, "Decentralized optimization, with application to multiple aircraft coordination," in *Decision and Control, 2002, Proceedings of the 41st IEEE Conference on*, vol. 1. IEEE, 2002, pp. 1147–1155.

[5] Y.-C. E. Yang, X. Cai, and D. M. Stipanović, "A decentralized optimization algorithm for multiagent system–based watershed management," *Water resources research*, vol. 45, no. 8, 2009.

[6] E. Loureiro, P. Nixon, and S. Dobson, "Adaptive management of shared resource pools with decentralized optimization and epidemics," in *Parallel, Distributed and Network-Based Processing (PDP), 2010 18th Euromicro International Conference on*. IEEE, 2010, pp. 51–58.

[7] P. Hu and X. Chen, "Decentralized ordinal optimization (doo) for networked systems," in *Automation Science and Engineering (CASE), 2015 IEEE International Conference on*. IEEE, 2015, pp. 787–792.

[8] A. Huang, S.-K. Joo, K.-B. Song, J.-H. Kim, and K. Lee, "Asynchronous decentralized method for interconnected electricity markets," *International Journal of Electrical Power & Energy Systems*, vol. 30, no. 4, pp. 283–290, 2008.

[9] D. Simon, *Evolutionary optimization algorithms*. John Wiley & Sons, 2013.

[10] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2001, vol. 2.

[11] E. Bengoetxea, P. Larrañaga, I. Bloch, and A. Perchant, "Estimation of distribution algorithms: A new evolutionary computation approach for graph matching problems," in *International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition*. Springer, 2001, pp. 454–469.