This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2703784, IEEE Transactions on Emerging Topics in Computing

High Performance Computing for Cyber Physical Social Systems by Using Evolutionary Multi-Objective Optimization Algorithm

Gai-Ge Wang, *Member, IEEE*, Xingjuan Cai, Zhihua Cui, *Member, IEEE*, Geyong Min, and Jinjun Chen, *Senior Member, IEEE*

***Abstract*—Cyber-physical social systems (CPSS) is an emerging complicated topic which is a combination of cyberspace, physical space, and social space. Many problems in CPSS can be mathematically modelled as optimization problems, and some of them are multi-objective optimization (MOO) problems (MOPs). In general, the MOPs are difficult to solve by traditional mathematical programming methods. High performance computing with much faster speed is required to address these issues. In this paper, a kind of high performance computing approaches, evolutionary multi-objective optimization (EMO) algorithms, is used to deal with these MOPs. A floorplanning case study is presented to demonstrate the feasibility of our proposed approach. B*-tree and a multistep simulated annealing (MSA) algorithm are cooperatively used to solve this case. As per experimental results for this case, the proposed method is well capable of searching for feasible floorplan solutions, and it can reach 74.44% (268/360) success rates for floorplanning problems.**

***Index Terms*—Cyber-physical social systems, High performance computing, Evolutionary multi-objective optimization, Floorplanning, Multistep simulated annealing.**

I. Introduction

THE focus of Internet of Things (IoT) or cyber-physical systems (CPS) is only linking the objects in physical space without the participation of human beings. While, cyber-physical social systems (CPSS) takes human social characteristics into account and can bridge the gap among cyberspace, physical space, and social space [1]. CPSS as a novel emerging paradigm has gained more and more attention from research and engineering community. In fact, CPSS integrates humans, computers, and things. Therefore, we can say, CPSS is the extension of IoT/CPS by incorporating human beings into it, and any technologies in IoT/CPS, socio-technical systems, and cyber-social systems are the part of CPSS [2].

CPSS as a novel emerging paradigm are still in their initial developing stage, therefore, no mature techniques to model and design them has been proposed up to now [3]. Zeng *et al*. proposed a system-level design methodology to model and design CPS, which allows refining high-level specifications into an underlying architecture leveraging platform-based technology [3]. Also, an intermediate representation model (IRM) is presented to capture the design requirements of CPS. Based on [3], an extended IRM is presented to adapt the modeling of multiple users for CPSS [4], which takes the social scenario of multiple users into account. Specifically, control flow is extended to model states and events of CPSS with the usage of a hierarchical Petri net [5]. However, these researches just deal with one issue for CPSS, and fail to deal with other methods. Also, when the complexity of problems of interest in CPSS becomes higher, it is difficult to be solved by the traditional mathematical programming algorithms. In this paper, a general-purpose high performance computing framework via evolutionary multi-objective optimization algorithm has been proposed to deal with these complicated problems.

Our main contributions can be provided below:

—The approach to model the problems in CPSS into an multi-objective optimization problem is proposed.

— A kind of high performance computing by using evolutionary multi-objective optimization algorithms is applied to solve the problem obtained before.

—A floorplanning case study is presented to demonstrate the feasibility of our proposed approach. Several variants of simulated annealing (SA) algorithms are cooperatively used to solve this case.

The rest of the paper is organized as follows. Section 2 represents some most representative work on cyber-physical social system. Several critical concepts of multi-objective

Manuscript received September 6, 2016; revised ****; accepted ****. This work was supported by the Natural Science Foundation of Jiangsu Province (No. BK20150239) and National Natural Science Foundation of China (No. 61503165, No. 61673196, and No. 61402207). This paper was recommended by ****. *(Corresponding author: Zhihua Cui.)*

G.-G. Wang is with the Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan, Shanxi, 030024, China; College of Information Science and Engineering, Ocean University of China, Qingdao, China; College of Computer, China University of Petroleum (Huadong), Qingdao, China; School of Computer Science, Jiangsu Normal University, Xuzhou, China; Institute of Algorithm and Big Data Analysis, and School of Computer, Northeast Normal University, Changchun, China (e-mail: gaigewang@gmail.com, gaigewang@163.com).

X. Cai and Z. Cui are with Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan, Shanxi, 030024, China (e-mail: xingjuancai@163.com, cuizhihua@gmail.com).

G. Min is with The University of Exeter, UK (G.Min@exeter.ac.uk).

J. Chen is with the Complex System and Computational Intelligence Laboratory, Taiyuan University of Science and Technology, Taiyuan, Shanxi, 030024, China; and also with Swinburne Data Science Research Institute, Swinburne University of Technology, Australia (e-mail: jinjun.chen@gmail.com).

optimization are provided in Section 3. In Section 4, a high performance computing technique by using evolutionary multi-objective optimization algorithm is firstly developed. In Section 5, a floorplanning case study by using B*-tree representation and a multistep simulated annealing (MSA) algorithm is presented to demonstrate the feasibility and effectiveness of our proposed framework. Finally, Section 6 concludes the work and suggests future work.

II. Related work

With the rapid development of CPSS technologies, mass emerging applications leverage such technologies in various domains. Typical applications of CPSS include smart home [6, 7], smart transportation system [8], smart medical service system [9], and smart cities [10-12]. In addition, the work in [13] can forecast bus arrival time via mobile phone. Leveraging passengers' participation and then sharing the information of bus location, data are transferred to a backend server, and then online data analysis and processing are executed to offer bus time services for users. In [14], a smart home cleaning application is developed based on CPSS. It can integrate a light-sensitive sensor and robot vacuum to accomplish automatic home cleaning. For a public space, the work in [15] gives a participatory sensing approach by handing out sensor probes for communities to assist in measuring exhaust, smog, chemical, noise, or dust. In addition, Campbell *et al.* [11] proposed an urban sensing architecture based on an opportunistic sensor network. It supports large-scale communication of public management and application in various domains.

Some applications are used to improve human life by the effective Human computer interaction (HCI) design. The work in [16] has developed a multimodal system for teaching blind handwriting. Using this system, the student can feel what the teacher draws on a tablet PC with a force-feedback pen that imitates the teacher's stylus movement. In [17], the authors have proposed a promising mobile interfaces design solution for the low-literacy populations. Three interface types are considered, including spoken dialog system, a graphical interface and a live operator.

Manifold CPSS devices have been designed in order to augment the experience of life. CPSS usually involve multiple elements, such as communicators, multimedia entertainment and business processing devices, etc. According to the characteristic of mobility of embedded devices, CPSS can be partitioned into the following categories: accompanied, portable, hand-held, wearable, implanted or imbedded [18]. Currently, many researchers have studied the device interoperability in CPSS. The work in [19] proposes an interoperability framework for smart home systems, which enables the integration of heterogeneous devices in smart home leveraging web services based technologies. However, such framework is quite limited and only appropriate for small scale application context. In [20], the authors present a middleware to provide a unified communication and collaboration mechanism for various devices under home automation environment. Furthermore, the middleware defined the notion of object proxy as a general solution to cope with everyday objects. Device management and discovery is a fundamental function for enabling the CPSS. Smart devices interoperation should be guaranteed to pave the way for the system development.

Even if these investigations are well applied in corresponding domains, all of them are application specific, and their design approaches are not scalable and generic to tailor it to the rapid development of CPSS.

Context awareness is a critical task for constructing the application of the CPSS [21]. Knowledge-based approaches have been employed in context awareness domain, such as ontology technologies and agent approaches. The work in [22] proposes the iConAwa context-aware system. Context is modeled in terms of ontology technology.

Context awareness plays a significant role in sensing the situation of CPSS. Particularly, the social context [23] is an emerging context that starts to gain more focus since CPS are extended into social domain and it will bring huge benefit for facilitating our life. It can be envisioned that further researches will place more emphasis on social awareness, which can be a critical issue in the future design of CPSS.

To address the design issues of CPSS, system-level design methodology offers a quite promising solution, which has been widely applied in the field of IoT or embedded systems. In [24], a system-level design method is proposed to address the design of a wearable system. The design process is the exploration of architecture in design space. System wearability and power consumption are quantified and used to evaluate design decisions. However, it can only address the design of single user and is hardly applied to social scenario. The work in [25] uses Y-chart design methodology to explore embedded system architectures. The system can be automatically generated by mapping from a high-level model to underlying architecture leveraging solving a multi-objective optimization problem. Actually, the method aims to design isolated embedded systems but fails to adapt the design to distributed systems. Besides, Malik *et al*. employ system-level language SystemJ to specify the computational and communicating portion of applications for distributed surveillance system [26]. The description can be used to generate executable codes. It offers an effective approach to model distributed and concurrency systems.

However, there is a limitation to apply them in CPSS. Most of them do not take the social effect into account, so they cannot satisfy the modeling and design needs within cyber, physical, and social space. Zeng *et al*. extend the system-level design method proposed by their previous work [3] and take the social impact into the design consideration to ensure nearly optimal design performance for CPSS [4].

Though the work above addressed the problem in CPSS in part, they fail to propose a general solution to solve most CPSS problems systematically. In this paper, we will propose a general technical framework with evolutionary multi-objective optimization algorithms to deal with the problems for CPSS.

III. Basic concepts

For single-objective optimization, it has only one global

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2703784, IEEE Transactions on Emerging Topics in Computing

optimum, therefore, it is much easier to handle and solve [27-30]. In addition, comparisons between different solutions can easily be done by the relational operators: >, ≥, <, ≤, and =. Because of unary property of such problems, the candidate solutions can be compared, and eventually provide the best solutions via a certain optimization technique. However, for multi-objective problems, solutions must be compared with more than one objective (criterion), which is more complicated than single-objective. We take minimum problem as an example, MOO can be formulated as follows:

$$\text{Minimize} \quad f(\mathbf{x}) = f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_l(\mathbf{x}) \tag{1}$$

$$\text{Subject to:} \begin{array}{l} g_i(x) \geq 0, i = 1, 2, \cdots, m \\ h_i(x) = 0, i = 1, 2, \cdots, p \\ L_i \leq x_i \leq U_i, i = 1, 2, \cdots, n \end{array} \tag{2}$$

where n, l, m, and p are the number of variables, objective functions, inequality constraints, and equality constraints, respectively. g_i and h_i indicate the *i-th* inequality and equality constraints, respectively, and [L_i, U_i] are the boundaries of the *i-th* variable.

It is obvious that, the solutions in multi-objective problems failed to be compared by using the relational operators mentioned above. Therefore, the relational operators must be extended for multi-objective problems. The four critical definitions in MOO are given as follows.

***Definition 1*. Pareto Dominance**

Assuming two vectors such as: $\mathbf{x}=(x_1, x_2, \cdots, x_k)$ and $\mathbf{y}=(y_1, y_2, \cdots, y_k)$. Vector $\mathbf{x}$ is said to dominate vector $\mathbf{y}$ (denote as $\mathbf{x} \prec \mathbf{y}$) if and only if:

$$\begin{array}{l} \forall i \in \{1, 2, \cdots, k\}: f(x_i) \leq f(y_i) \\ \wedge \exists i \in \{1, 2, \cdots, k\}: f(x_i) < f(y_i) \end{array} \tag{3}$$

The definition of Pareto optimality is as follows [31]:

***Definition 2*. Pareto Optimality**

A solution $\mathbf{x} \in \mathbf{X}$ is called Pareto-optimal if and only if:

$$\nexists y \in X \mid f(\mathbf{y}) \prec f(\mathbf{x}) \tag{4}$$

***Definition 3*. Pareto optimal set**

The set all Pareto-optimal solutions is called Pareto set as follows:

$$P_s := \{\mathbf{x}, \mathbf{y} \in \mathbf{X} \mid \exists f(\mathbf{y}) \succ f(\mathbf{x})\} \tag{5}$$

***Definition 4*. Pareto optimal front**

A set containing the value of objective functions for Pareto solutions set:

$$P_f := \{f(\mathbf{x}) \mid \mathbf{x} \in P_s\} \tag{6}$$

For solving a MOP, we have to find Pareto optimal set, which is the set of solutions representing the best trade-offs between different objectives. In recent years, multi-objective optimization algorithms have been used in many fields, such as big data [32], scheduling [33, 34], data downloading [35], power flow problem [36], vehicle routing problem [37], wireless sensor networks [38].

IV. High Performance Computing for CPSS by Using EMO Algorithm

In this section, a general framework that shows how to solve the problems in CPSS with high performance computing techniques by using evolutionary multi-objective optimization algorithms is presented.

Though different problems have different properties, forms and originations, they can be mathematically modelled into a single-objective/multi-objective optimization (SOO/MOO) problems (SOPs/MOPs). Clearly, SOO/SOP can be considered as a special case of MOO/MOP. Therefore, in the following, we only take MOO/MOP into account to express our proposed techniques.

Firstly, the problem for CPSS must be transferred into a mathematical problem. Though the originated problems have different forms, the transferred problems can be expressed in Eqs. (1)-(2), and they vary according to the original problems (e.g., triple-objective integer linear optimization (ILP) [3], and triple-objective combinatorial optimization problem [4]). The variables in equations (1) and (2) may include integers, real numbers, complex, or their combinations. Through modelling, we get a standard MOP that will be further addressed later.

In general, the MOP in equations (1) and (2), is a high dimensional complicated problem, so it is hard to be solved by traditional mathematical methods. So, a certain kind of high performance computing techniques must be used. These kinds of high performance computing techniques can be implemented in parallel [39]. One of the most representative paradigms is evolutionary algorithms. Because they have parallel properties in nature, they have gained popularity within the research community and industry [40]. Here, we use multi-objective version of evolutionary algorithms, called evolutionary multi-objective optimization (EMO) algorithms, to solve MOPs.

In this paper, B*-tree and a MSA algorithm are combined to solve a CPSS problem. The detailed description can be found in Section 5.

V. Simulation Results

As mentioned before, CPSS involves cyberspace, physical space, and social space [1]. The floorplan is one of the most representative problems in the research of CPSS, which has been used in many literatures [3, 4]. In this paper, we use a kind of floorplan problem in the field of VLSI (very-large-scale integration) to demonstrate the feasibility and effectiveness of our proposed technical framework for the problems in CPSS. B*-tree and a MSA algorithm are cooperatively used to solve this case.

A. Problem description

If we have some blocks with the total area A and the maximum allowed percent of dead space Γ, we must put these blocks into a fixed outline and its aspect ratio is denoted as R. Its height and width are respectively represented by H_0, and W_0. Accordingly,

$$H_0 W_0 = (1+\Gamma)A \tag{7}$$

$$R = H_0 / W_0 \tag{8}$$

According to Eqs. (7)-(8), we can get H_0 and W_0

$$H_0 = \sqrt{(1+\Gamma)AR} \tag{9}$$

$$W_0 = \sqrt{(1+\Gamma)A/R} \tag{10}$$

For this problem, both the area and wirelength will be minimized. Therefore, it can be modeled into the MOPs below

$$\begin{aligned} &\min\ f_1 = A_c \\ &\min\ f_2 = W_c \\ &s.t. \begin{cases} W \le W_0 \\ H \le H_0 \end{cases} \end{aligned} \tag{11}$$

where A_c and W_c are the area and wirelength of a floorplan, respectively. W and H denote the current width and height of the floorplan, respectively. W_0 and H_0 reprent the maximum allowed width and height, respectively.

B. Cost functions

Here, we will take the simplest fixed-outline floorplanning as an example to give the cost function. Several representative mathematical models have been proposed by many researchers, as shown below.

1) Cost function based on height and width violations

Adya and Markov [41] proposed a cost function, which can be give below

$$\begin{aligned} Cost_1 = &\max(H-H_0,0)+ \\ &\max(W-W_0,0) \end{aligned} \tag{12}$$

2) Cost function based on the maximum value of height and width violations

In addition, Adya and Markov [41] proposed the second one by using the maximum value of height and width violations, which can be given below

$$Cost_2 = \max(H-H_0, W-W_0) \tag{13}$$

It can be seen that, this cost function may lead to oscillation, which will hinder the convergence sometimes. Especially, its results have confliction with the previous one.

3) Cost function based on normalized height and width violations

In order to make a better tradeoff between the flexibility of the small and large blocks, Liu *et al.* [42] proposed another cost function by introducing additional weights into (12). Additionally, this function involves the number of blocks violating outline limit. Accordingly, the modified cost function can be expressed as

$$\begin{aligned} Cost_1 = &\max(H-H_0,0)/H_0 + \\ &\max(W-W_0,0)/W_0 + N_{vio\_b} \end{aligned} \tag{14}$$

where N_{vio_b} is the number of blocks which violating the outline constraints.

4) Cost function based on height & width violations and aspect ratio

Chen and Yoshimura [43] proposed another cost function by introducing aspect ratio λ, which can be given as

$$\begin{aligned} Cost_4 = &E_W/\sqrt{\lambda} + E_H \cdot \sqrt{\lambda} + \\ &C_1 \cdot \max(E_W/\sqrt{\lambda}, E_H \cdot \sqrt{\lambda}) + \\ &C_2 \cdot \max(W/\sqrt{\lambda}, H\sqrt{\lambda}) \end{aligned} \tag{15}$$

where $E_W = \max(W-W_0,0)$ and $E_H = \max(H-H_0,0)$ are the excessive width and height of the floorplan, respectively. The coefficients C_1 and C_2 are set to 1 and 1/16, respectively. In order to comply with the aspect ratio R, $1/R$ is used to replace λ. Therefore, Eq. (15) can be given as

$$\begin{aligned} Cost_4 = &E_W \cdot \sqrt{R} + E_H/\sqrt{R} + \\ &C_1 \cdot \max(E_W \cdot \sqrt{R}, E_H/\sqrt{R}) + \\ &C_2 \cdot \max(W \cdot \sqrt{R}, H/\sqrt{R}) \end{aligned} \tag{16}$$

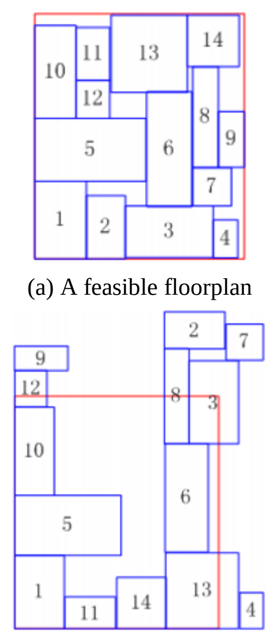
5) Two More Cost functions

For the four models mentioned before, their basis is excessive width&height limits for fixed-outline floorplanning. In order to model the fixed-outline floorplanning more accurately, two more modified cost functions are given as follows.

$$\begin{aligned} Cost_5 = &\sum_{i \in S_H} \max(y_i - H_0, 0)/H_0 + \\ &\sum_{j \in S_W} \max(x_j - W_0, 0)/W_0 \end{aligned} \tag{17}$$

$$\begin{aligned} Cost_6 = &\sum_{i \in S_H} \max(y_i - H_0, 0)/H_0 + \\ &\sum_{j \in S_W} \max(x_j - W_0, 0)/W_0 + N_{vio\_b} \end{aligned} \tag{18}$$

where S_H and S_W represent the set of height, and width violations, respectively. *i* represents the index of block *i* that is out of the height limit. y_i and x_j are the y-coordinate and x-coordinate of the upper right corner of block i and block *j*,



(a) A feasible floorplan

(b) An infeasible floorplan

Fig. 1. Two kinds of floorplans, the first is a feasible solution, and the second violates the fixed-outline constraints.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TETC.2017.2703784, IEEE Transactions on Emerging Topics in Computing

respectively. The instances of the excessive width&height terms are illustrated, as shown in Fig. 1.

According to Eq. (17), the cost function value is $Cost_5=\sum_{i=2,3,7,8,9,12}\max(y_i-H_0,0)/H_0+\sum_{j=2,3,4,7,13}\max(x_j-W_0,0)/W_0$, in the meantime, the cost function value is equal to $Cost_6= Cost_5+8$ in terms of (12). Concretely, blocks 8, 9 and 12 violate the height limit, blocks 4 and 13 violate the width limit, and blocks 2, 3 and 7 violate height and width limits simultaneously.

The six cost functions are only used in simplest condition. For more complicated case, such as the chip area, wirelength and fixed-outline constraints simultaneously, a penalty function combining A_c, W_c, H_0 and W_0, is remodeled as

$$f_i=\alpha\cdot\frac{A_c}{A_a}+(1-\alpha)\cdot\frac{W_c}{W_a}+\theta\cdot Cost_i \tag{19}$$

where A_a and W_a computed by averaging the areas and wirelengths are the average area, and wirelength, respectively, which are . α is equal to 0.5 controlling the weights for area and wirelength. The value m is set to 30 depending on the number of blocks. f_i is the objective function which combines area, wirelength and $Cost_i$. θ is called penalty factor. θ is set to 10^6 for all test instances in this paper.

C. Multistep SA algorithm

For the basic SA, in the cooling schedule, temperature drop is implemented multiplying T by the factor r ($r<1$) at each iteration. Initially, suppose the temperature $T_0=10^6$

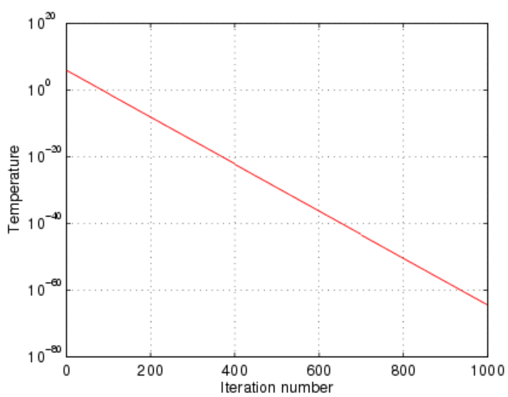


Fig. 2. The evolutionary curve of the temperature *T*.

which is suitable for most problems, the factor $r=0.85$, by using the above cooling schedule, an evolutionary curve of the temperature versus iteration number is shown in Fig. 2.

From Fig. 2, we can see the temperature T decreases quickly with the increment of iteration. Suppose the range of the increment of cost function value ΔC is $[10^{-5},10^5]$, the rate of accepting a worse solution is close 0 fast at 14th iteration for $\Delta C=10^{-5}$, and it is reduced sharply for $\Delta C=10^5$. In other words, some promising worse solutions cannot be utilized to find better ones during the middle and later stages of SA process. This predicament isn't beneficial for helping the trapped solutions to get rid of the local optimums. Thus, it is necessary to advance a higher performance of SA by introducing some other useful strategies.

In order to overcome premature convergence, a multistep SA (MSA) algorithm is developed in this paper, and it can be represented below: It has a small probability that a trapped solution will escape from the local solution since SA is similar to greedy algorithm in the later optimization process. Meanwhile, it is impossible to search for a better solution only through one independent perturbation. Therefore, a multistep method is incorporated into the basic SA with the aim of improving the usage of worse solutions.

If MSA fails to find better solution many times, it is unnecessary to repeat the basic optimization process of SA. It is high exact time that we should turn to other strategy. In this paper, a multistep strategy is used to overcome this problem. In MSA, it can perturb a trapped solution many times in a continuous way. The framework of MSA can be found in Fig. 3.

Line Procedure of MSA

```
Initialize a floorplan F, and F_best=F;
Initialize temperature T0 (T0>0) and r (r<1);
Set parameters N_fail=0, U_fail=10n, N_c=20, N_s=3;
While the "frozen" state is not reached
  For k=1 to k_max
    Perturb F to obtain a new one F_new;
    Calculate △C=cost(F_new)-cost(F);
    If △C <0
      F=F_new;
      If cost(F_new)<cost(F_best)
        F_best = F_new;
      End If
      N_fail=0;
    Else
      If r_n<exp(-△C /T)
        F= F_new;
      Else
        Exclude F_new, and save F; % up-hill move
      End If
      N_fail= N_fail +1;
    End If
    If N_fail = U_fail
      Obtain current floorplan F, and set sig=0;
      For i=1 to N_c
        If sig=0
          Save F;
```

```
            End If
            Randomly select a block b_p from F;
            For j=1 to N_s
              Perturb b_p to generate a new one F_new
            If F_new is better than F
              F= F_new, set sig=1, and break;
            Else
              sig=0;
            End If
          End For
        End For
        If sig=0
          Save F;
        End If
        N_fail =0;
        End If
      End For
      T=rT; % decrease temperature
    End While
    Return F_best
```

Fig. 3 Floorplanning based on MSA

In Fig. 3, N_{fail} represents the number of successive failures, U_{fail}=10n is stagnation period, and n is the number of blocks. Here, a multistep strategy will be implemented if N_{fail} increases to the allowed upper limit U_{fail}=10n. In MSA, N_c denotes the runs of multistep search, and N_s is the maximum allowed number of successive perturbations for block b_p (p=1, 2, …, n).

D. Experimental results and analysis

Here, six algorithms are used to address the floorplanning problems only considering fixed outline constraints. The first four algorithms are designed according to cost functions $Cost_i$ (i=1,2,3,4) [41-43] and SA, and the last two ones are designed according to $Cost_i$ (i=5,6) and MSA. For simplicity, the remarks SA-C1, SA-C2, SA-C3, SA-C4, MSA-C5 and MSA-C6 are used be short for these six approaches, respectively. For each instance, the maximum allowed percentage of dead space is set to $\Gamma = 15\%$ and $\Gamma = 10\%$, respectively. In addition, five different aspect ratios (R=1.0, 1.5, 2.0, 2.5, 3.0) are used in our experiments. The maximal runtime of the six algorithms mentioned before is set to 5s, 5s, 10s, 15s, 20s, and 30s, respectively, for n10, n30, n50, n100, n200, and n300 with $\Gamma = 15\%$; for n10, n30, n50, n100, n200, and n300 with $\Gamma = 10\%$, the maximal allowed runtime is set to 20s, 20s, 40s, 60s, 80s, and 120s, respectively. Moreover, if an algorithm fails to find the final best solution within the outline constraints in a fixed time limit, it will return failure. B*-tree and SA (MSA) are cooperatively used to implement the experiments. The initial temperature $T_0 = \Delta_{avg} / P$ [44], Δ_{avg} is the average uphill cost, and P is the initial probability to accept uphill solutions. The experiments are implemented by C++ with Intel Core i5-2410M CPU@2.30 GHz. 20 independent runs are implemented in order to get the most representative statistical results, as shown in Tables I-II.

TABLE I
SUCCESS RATES AND RUNTIME OF SIX APPROACHES FOR SIX CIRCUITS WITH DIFFERENT ASPECT RATIOS (THE MAXIMUM ALLOWED PERCENTAGE OF DEAD SPACE Γ=15%)

Circuits	R	SA-C1		SA-C2		SA-C3		SA-C4		MSA-C5		MSA-C6	
		SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)
n10	1.0	100	0.023	100	0.03	100	0.01	95	0.275	100	0.022	100	0.023
	1.5	100	0.027	100	0.019	100	0.011	95	0.281	100	0.019	100	0.024
	2.0	100	0.143	100	0.015	100	0.269	100	0.019	100	0.036	100	0.214
	2.5	100	0.011	100	0.023	100	0.08	95	0.268	100	0.014	100	0.084
	3.0	95	0.292	95	0.278	65	1.763	80	1.016	100	0.059	100	0.026
n30	1.0	100	0.095	100	0.1	100	0.077	100	0.08	100	0.077	100	0.085
	1.5	100	0.135	100	0.096	100	0.118	100	0.087	100	0.071	100	0.079
	2.0	100	0.15	100	0.105	100	0.093	100	0.081	100	0.046	100	0.072
	2.5	100	0.149	100	0.106	100	0.116	100	0.126	100	0.082	100	0.047
	3.0	100	0.162	100	0.122	100	0.062	100	0.123	100	0.082	100	0.092
n50	1.0	100	0.216	100	0.246	100	0.166	100	0.211	100	0.165	100	0.2

	1.5	100	0.286	100	0.285	100	0.205	100	0.23	100	0.151	100	0.203
	2.0	100	0.455	100	0.324	100	0.252	100	0.333	100	0.175	100	0.207
	2.5	100	0.465	100	0.319	100	0.327	100	0.327	100	0.212	100	0.256
	3.0	100	0.475	100	0.387	100	0.316	100	0.385	100	0.159	100	0.157
n100	1.0	100	0.945	100	1.392	100	0.856	100	1.068	100	0.693	100	1.024
	1.5	100	2.044	100	1.688	100	1.168	100	1.68	100	0.823	100	0.962
	2.0	100	2.34	100	1.945	100	1.582	100	1.88	100	0.751	100	0.84
	2.5	100	2.944	100	2.647	100	1.306	100	2.117	100	0.833	100	0.86
	3.0	100	3.234	100	2.291	100	1.416	100	2.575	100	0.816	100	1.01
n200	1.0	100	5.263	100	10.762	100	4.874	100	5.952	100	3.564	100	4.287
	1.5	100	5.964	100	11.581	100	3.131	100	6.46	100	3.545	100	4.172
	2.0	60	14.229	80	16.229	100	5.125	85	14.801	100	3.53	100	4.139
	2.5	35	16.165	40	18.433	100	5.98	60	14.585	100	3.403	100	4.203
	3.0	25	19.637	30	19.141	100	7.66	50	17.783	100	3.29	100	3.91
n300	1.0	100	9.677	30	28.809	100	10.138	90	14.777	100	10.035	100	10.925
	1.5	90	12.221	0	30	100	9.625	75	16.755	100	9.58	100	10.858
	2.0	45	23.312	0	30	100	11.54	55	23.19	100	9.047	100	12.192
	2.5	15	28.823	0	30	100	12.49	45	22.902	100	9.712	100	10.647
	3.0	10	29.357	0	30	100	13.666	25	26.605	100	11.058	100	11.381

TABLE II
Success rates and runtime of six approaches for six circuits with different aspect ratios (The maximum allowed percentage of dead space Γ=10%)

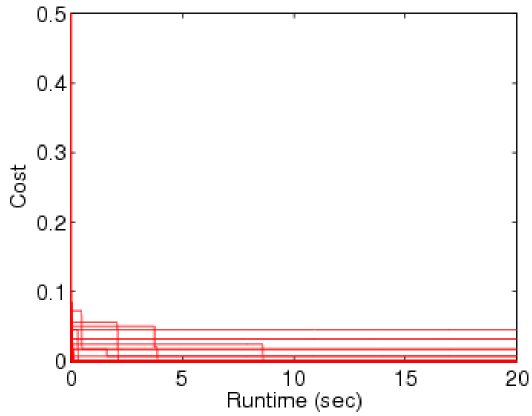
Circuits	R	SA-C1		SA-C2		SA-C3		SA-C4		MSA-C5		MSA-C6	
		SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)	SR(%)	ART(s)
n10	1.0	40	12.05	80	4.17	55	9.12	50	10.06	90	2.59	80	4.52
	1.5	75	5.08	65	7.04	70	6.15	65	7.38	85	3.41	95	1.33
	2.0	75	5.11	70	6.03	80	4.4	75	5.04	80	4.05	80	4.11
	2.5	85	3.49	60	8.43	85	3.09	70	6.24	75	5.12	75	5.11
	3.0	60	8.33	70	6.1	45	11.08	55	9.08	80	4.65	70	7.49
n30	1.0	100	0.21	100	0.27	100	0.39	100	0.28	100	0.17	100	0.31
	1.5	100	0.34	100	0.25	100	0.45	100	0.34	100	0.22	100	0.28
	2.0	100	0.39	100	0.26	100	0.55	100	0.31	100	0.25	100	0.31
	2.5	100	0.39	100	0.3	100	0.52	100	0.28	100	0.33	100	0.3
	3.0	100	0.43	100	0.3	100	0.59	100	0.3	100	0.3	100	0.42
n50	1.0	100	0.71	100	0.75	100	1.07	100	0.54	100	0.43	100	0.66
	1.5	100	0.89	100	0.85	100	1.06	100	0.94	100	0.35	100	0.6
	2.0	100	0.98	100	1.29	100	1.07	100	0.77	100	0.52	100	0.85
	2.5	100	1.02	100	0.94	100	0.96	100	0.94	100	0.51	100	0.97
	3.0	100	1.3	100	1.04	100	0.98	100	0.87	100	0.46	100	0.63
n100	1.0	100	2.48	100	4.53	100	6.26	100	3.25	100	0.46	100	3.3
	1.5	100	5.67	100	6.31	100	6.05	100	3.98	100	1.77	100	3.49

	2.0	100	7	100	5.95	100	5.09	100	5.37	100	1.91	100	3.09
	2.5	100	8.35	100	6.67	100	5.3	100	6.69	100	1.86	100	3.17
	3.0	100	9.26	100	9.28	100	7.6	100	7.32	100	2.87	100	3.48
n200	1.0	100	11.56	75	65.83	100	32.59	100	18.69	100	10.06	100	18.07
	1.5	90	30.15	60	69.17	100	39.43	95	42.65	100	9.16	100	15.76
	2.0	95	50.5	35	76.15	100	36.71	90	46.25	100	10.16	100	14.02
	2.5	80	58.12	40	75.2	100	34.61	75	47.92	100	7.83	100	13.42
	3.0	5	79.59	5	79.57	100	41.46	45	69.99	100	8.19	100	8.96
n300	1.0	100	34.01	0	120	90	85.7	90	54.37	100	30.02	100	39.17
	1.5	80	57.44	0	120	80	96.48	60	75.94	100	27.91	100	34.03
	2.0	90	49.43	0	120	80	90.08	50	89.78	100	28.05	100	29.13
	2.5	0	120	0	120	65	101.1	25	101.49	100	19.31	100	25.62
	3.0	0	120	0	120	60	101.36	25	96.15	100	19.21	100	21.83

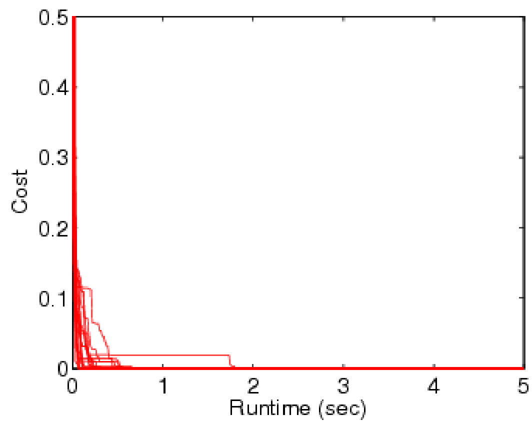
From Table I, *R* is aspect ratio (As Eq. (8)), the term "SR" represents success rate, and "ART" represents average runtime. For n10 and *R*=1.0, 1.5, 2.0, 2.5, three algorithms (SA-C1, SA-C2 and SA-C3) reach a 100% success rate; meanwhile, two algorithms (SA-C1 and SA-C2) cannot find the final solution only once for *R*=3.0; while SA-C3 fails triple for *R*=3.0. Moreover, SA-C4 can be succeed in reaching a 100% success rate only for *R*=2.0. Comparing with four algorithms above, two variants of MSA algorithms (MSA-C5 and MSA-C6) are well capable of getting a 100% success rate for all the five aspect ratios. For n30, n50 and n100, all six algorithms perform well and all of them are successful in reaching a 100% success rate for any aspect ratio, and MSA-C5 has a fastest speed for most instances. For n200, three algorithms (SA-C1, SA-C2 and SA-C4) perform well and are well capable of obtaining a 100% success rate for *R*=1.0, 1.5, respectively, but they have no ability of finding the feasible solution several times for *R*=2.0, 2.5, and 3.0. Other three algorithms (SA-C3, MSA-C5 and MSA-C6) are able to obtain a 100% success rate under any condition, and MSA-C5 has a fastest speed among them for most instances. For n300 and *R*=1.5, 2.0, 2.5, 3.0, SA-C2 performs not well and its SRs are equal to 0. For SA-C1 and SA-C4, their success rates decrease with the increment of *R*. In the meantime, three algorithms (SA-C3, MSA-C5 and MSA-C6) are still able to find the best final solutions with a 100% success rate, and MSA-C5 performs fastest for all five aspect ratios.

From Table II, for n10, no one algorithm can find the final best solution. For n30, n50 and n100, all the algorithms perform well and can find the final best solutions with a 100% success rate, and their ARTs are small in most cases. For n200, SA-C3, MSA-C5 and MSA-C6 are able to find feasible solutions with a 100% success rate. MSA-C5 has a 3.24×, 4.3×, 3.61×, 4.42× and 5.06× times speed than SA-C3 when finding feasible solutions for n200 with *R*=1.0, 1.5, 2.0, 2.5 and 3.0. In addition, MSA-C6 performs faster than SA-C3 for n200. Moreover, for SA-C1, SA-C2 and SA-C4, their success rates decrease with the increment of aspect ratio, and regarding average runtime, they take more time than MSA-C5 for most problems. For n300, SA-C1 cannot find a best final floorplan for both *R*=2.5 and *R*=3.0 regarding "SR", and SA-C2 doesn't search for the best solution for *R*=1.0, 1.5, 2.0, 2.5 and 3.0. Meanwhile, for SA-C3, its average runtime is bigger than SA-C4 for n300, but its success rates are higher than SA-C4 in most cases. In addition, MSA-C5 is well capable of finding the feasible floorplan solutions for n300. MSA-C5 has a 2.85×, 3.46×, 3.21×, 5.24× and 5.28× times speedups than SA-C3 for n300 with *R*=1.0, 1.5, 2.0, 2.5 and 3.0. in addition, for MSA-C5, its success rate is higher than SA-C1 for n300 under any condition. Except MSA-C5, MSA-C6 performs better than the first four algorithms for n300 under any conditions. In sum, two variants of MSA (MSA-C5 and MSA-C6) have better performance than four versions of SAs (SA-C1, SA-C2, SA-C3 and SA-C4) when addressing the fixed-outline constraints. Especially, MSA-C5 has a little better performance than MSA-C6 for most instances according to "ART".

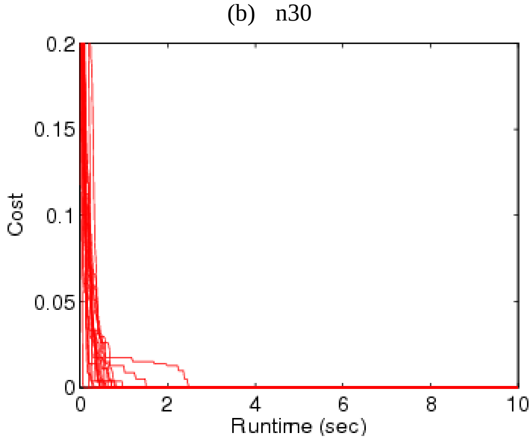
The convergent history of MSA-C5 for six floorplanning problems is also illustrated, as shown in Fig. 4. In Fig. 4, the aspect ratio and the maximum allowed percentage of dead space are set to *R*=3.0 and $\Gamma = 10\%$, respectively, for the circuits n10, n30, n50, n100, n200 and n300. Because of lack of flexibility of blocks in n10, MSA-C5 cannot find the feasible solutions many times. In addition, clearly, MSA-C5 has a fast convergent speed at the begin of optimization process, and becomes more and more slow as iteration goes. At last, it can get the feasible solutions, which shows the effectiveness of MSA-C5 when dealing with floorplan problems.



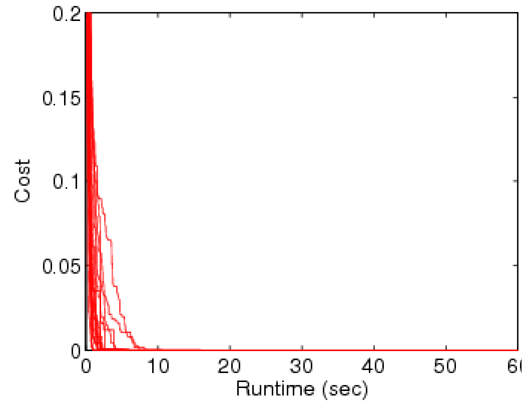
(a) n10



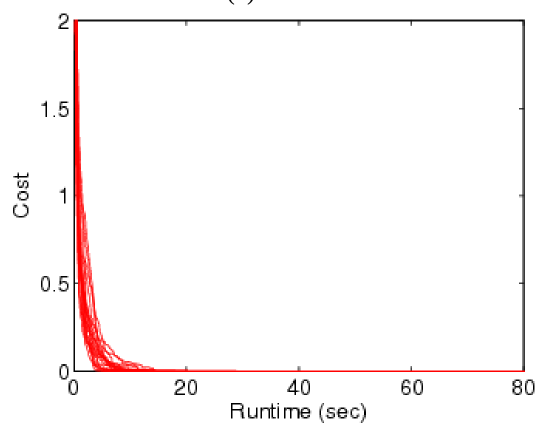
(b) n30



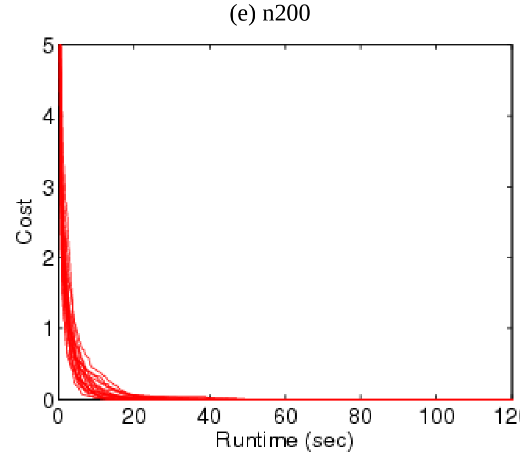
(c) n50



(d) n100

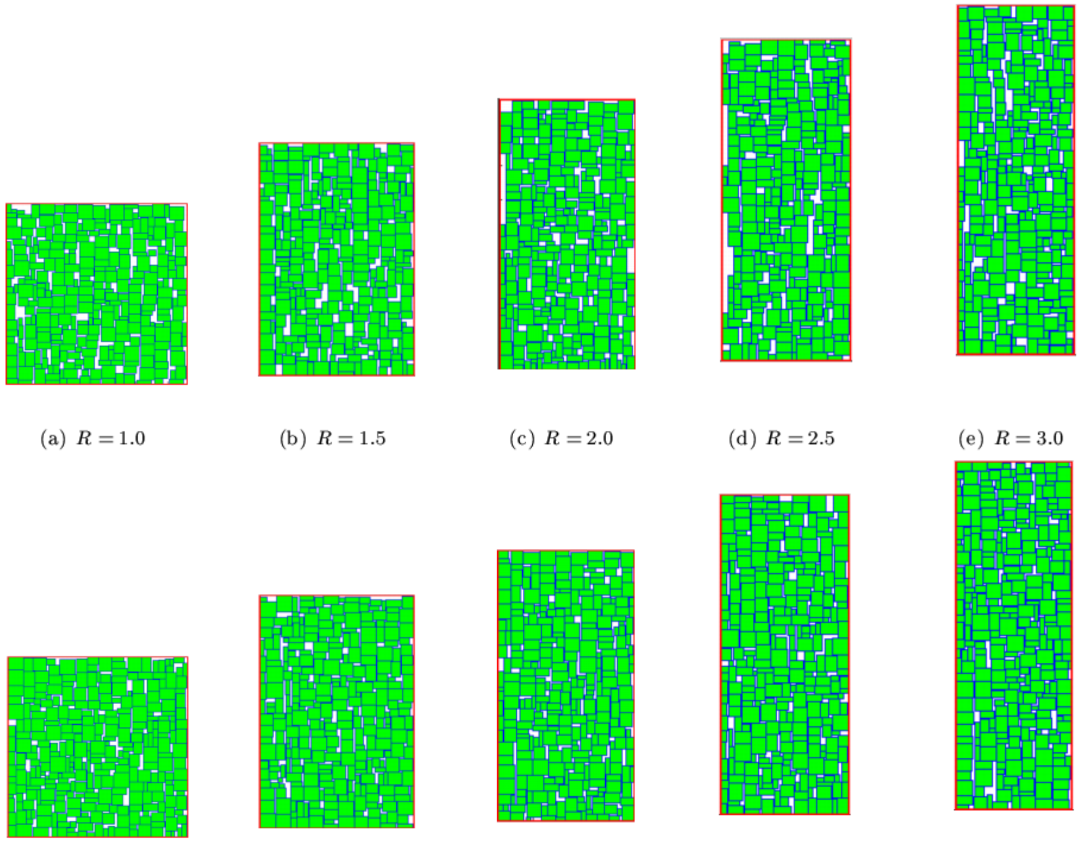


(e) n200



(f) n300

Fig. 4 Convergence curves of MSA for six floorplanning problems using the area violation model (Aspect ratio R=3.0, and the maximum allowed percentage of dead space Γ=10%))



(a) $R = 1.0$ (b) $R = 1.5$ (c) $R = 2.0$ (d) $R = 2.5$ (e) $R = 3.0$

(f) $R = 1.0$ (g) $R = 1.5$ (h) $R = 2.0$ (i) $R = 2.5$ (j) $R = 3.0$

Fig. 5 Ten floorplans of n300 obtained by MSA-C5. The maximum allowed percentage of dead space is Γ=15% for (a)-(e), and Γ=10% for (f)-(j)

In addition, ten floorplans of n300 found by MSA-C5 are also illustrated in Fig. 5. The maximum allowed percentage of dead space of the first five floorplans is $\Gamma = 15\%$ (Figs. 5(a)-(e)). In order to get more compact floorplans, we lower Γ to 10% from 15%. From Figs. 5(f)-(j), the area of the dead space of the second five floorplans is less than the first five floorplans. With the decrement of Γ, the flexibility of the blocks significantly decreases. So, more time will be used in order to adjust the positions of blocks. In summation, MSA-C5 is an efficient promising algorithm when addressing floorplanning problem. This also indicate, our proposed technical framework can solve the problem in CPSS well.

VI. Conclusion

In this paper, a general technical framework for CPSS is proposed. The approach to model the problems in CPSS into a multi-objective optimization problem is presented, which is generally a complicated one. High performance computing with much faster speed is required to address these issues. A kind of high performance computing by using EMO algorithms is applied to solve the problem. Finally, a floorplanning case study is presented to demonstrate effectiveness of our proposed technical framework. B*-tree and a MSA algorithm as an evolutionary multi-objective optimization tool are cooperatively used to solve this case.

Though our proposed algorithms can well solve the problems, in future, the following points should be further addressed.

Firstly, in this paper, we just use an improved simulated annealing as an example to verify our proposed general-purpose framework. For the other multi-objective optimization algorithms, such as NSGA-II, HypE, MOPSO, and MOEA/D, we will use them to solve the problem in our future research.

Secondly, when the number of objectives is more than three, we can call it many-objective optimization. In essence, multi-objective optimization can be considered as a special case of many-objective optimization. If we can model the problems in CPSS into many-objective problems, many-objective optimization algorithms (like NSGA-III) can be surely used to solve these problems. This will be our next research point.

At last, in this paper, we use a floorplan problem to verify and demonstrate the effectiveness and efficiency of our proposed methods. In future, other problems in CPSS will be used to further verify our proposed method.

References

[1] F.-Y. Wang, "The Emergence of Intelligent Enterprises: From CPS to CPSS," *IEEE Intelligent Systems*, vol. 25, no. 4, pp. 85-88, 2010.

[2] A. Sheth, P. Anantharam, and C. Henson, "Physical-Cyber-Social Computing: An Early 21st Century Approach," *IEEE Intelligent Systems*, vol. 28, no. 1, pp. 78-82, 2013.

[3] J. Zeng, L. Yang, J. Ma, and M. Guo, "HyperspaceFlow: A System-Level Design Methodology for Smart Space," *IEEE*

Transactions on Emerging Topics in Computing, vol. 4, no. 4, pp. 568-583, 2016.

[4] J. Zeng, L. T. Yang, and J. Ma, "A System-Level Modeling and Design for Cyber-Physical-Social Systems," *ACM Transactions on Embedded Computing Systems,* vol. 15, no. 2, pp. 1-26, 2016.

[5] R. Fehling, "A concept of hierarchical Petri nets with building blocks," *Advances in Petri Nets 1993*, G. Rozenberg, ed., pp. 148-168, Berlin, Heidelberg: Springer Berlin Heidelberg, 1993.

[6] D. J. Cook, A. S. Crandall, B. L. Thomas, and N. C. Krishnan, "CASAS: A Smart Home in a Box," *Computer,* vol. 46, no. 7, pp. 62-69, 2013.

[7] Y.-W. Kao, and S.-M. Yuan, "User-configurable semantic home automation," *Computer Standards & Interfaces,* vol. 34, no. 1, pp. 171-188, 2012.

[8] Y. Jing, Z. Yu, X. Xing, and S. Guangzhong, "T-Drive: Enhancing Driving Directions with Taxi Drivers' Intelligence," *IEEE Transactions on Knowledge and Data Engineering,* vol. 25, no. 1, pp. 220-232, 2013.

[9] M. Alwan, S. Dalal, D. Mack, S. Kell, B. Turner, J. Leachtenauer, and R. Felder, "Impact of Monitoring Technology in Assisted Living: Outcome Pilot," *IEEE Transactions on Information Technology in Biomedicine,* vol. 10, no. 1, pp. 192-198, 2006.

[10] Y. Wang, L. Wei, A. V. Vasilakos, and Q. Jin, "Device-to-Device based mobile social networking in proximity (MSNP) on smartphones: Framework, challenges and prototype," *Future Generation Computer Systems*, 2015.

[11] A. T. Campbell, S. B. Eisenman, N. D. Lane, E. Miluzzo, R. A. Peterson, H. Lu, X. Zheng, M. Musolesi, K. Fodor, and G.-S. Ahn, "The Rise of People-Centric Sensing," *IEEE Internet Computing,* vol. 12, no. 4, pp. 12-21, 2008.

[12] F. Calabrese, and C. Ratti, "Real time rome," *Networks and Communication studies,* vol. 20, no. 3-4, pp. 247-258, 2006.

[13] P. Zhou, Y. Zheng, and M. Li, "How long to wait? Predicting bus arrival time with mobile phone based participatory sensing," in Proceedings of the 10th International Conference on Mobile systems, Applications, and Services, Low Wood Bay, Lake District, UK, 2012, pp. 379-392.

[14] A. Smirnov, A. Kashevnik, and A. Ponomarev, "Multi-level Self-organization in Cyber-Physical-Social Systems: Smart Home Cleaning Scenario," *Procedia CIRP,* vol. 30, pp. 329-334, 2015.

[15] S. Kuznetsov, and E. Paulos, "Participatory sensing in public spaces: activating urban surfaces with sensor probes," in Proceedings of the 8th ACM Conference on Designing Interactive Systems, Aarhus, Denmark, 2010, pp. 21-30.

[16] B. Plimmer, P. Reid, R. Blagojevic, A. Crossan, and S. Brewster, "Signing on the tactile line," *ACM Transactions on Computer-Human Interaction,* vol. 18, no. 3, pp. 1-29, 2011.

[17] I. Medhi, S. Patnaik, E. Brunskill, S. N. N. Gautama, W. Thies, and K. Toyama, "Designing mobile interfaces for novice and low-literacy users," *ACM Transactions on Computer-Human Interaction,* vol. 18, no. 1, pp. 1-28, 2011.

[18] A. A. Berlin, and K. J. Gabriel, "Distributed MEMS: New Challenges for Computation," *IEEE Computational Science & Engineering,* vol. 4, no. 1, pp. 12-16, 1997.

[19] T. Perumal, A. R. Ramli, and C. Y. Leong, "Interoperability framework for smart home systems," *IEEE Transactions on Consumer Electronics,* vol. 57, no. 4, pp. 1607-1611, 2011.

[20] P. Lukowicz, S. Pentland, and A. Ferscha, "From Context Awareness to Socially Aware Computing," *IEEE Pervasive Computing,* vol. 11, no. 1, pp. 32-41, 2012.

[21] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, "A survey of context modelling and reasoning techniques," *Pervasive and Mobile Computing,* vol. 6, no. 2, pp. 161-180, 2010.

[22] Ö. Yılmaz, and R. C. Erdur, "iConAwa – An intelligent context-aware system," *Expert Systems with Applications,* vol. 39, no. 3, pp. 2907-2918, 2012.

[23] G. Liang, and J. Cao, "Social context-aware middleware: A survey," *Pervasive and Mobile Computing,* vol. 17, pp. 207-219, 2015.

[24] U. Anliker, J. Beutel, M. Dyer, R. Enzler, P. Lukowicz, L. Thiele, and G. Troster, "A systematic approach to the design of distributed wearable systems," *IEEE Transactions on Computers,* vol. 53, no. 8, pp. 1017-1033, 2004.

[25] A. D. Pimentel, C. Erbas, and S. Polstra, "A systematic approach to exploring embedded system architectures at multiple abstraction levels," *IEEE Transactions on Computers,* vol. 55, no. 2, pp. 99-112, 2006.

[26] A. Malik, Z. Salcic, C. Chong, and S. Javed, "System-level approach to the design of a smart distributed surveillance system using systemj," *ACM Transactions on Embedded Computing Systems,* vol. 11, no. 4, pp. 1-24, 2012.

[27] X. Cai, X.-z. Gao, and Y. Xue, "Improved bat algorithm with optimal forage strategy and random disturbance strategy," *International Journal of Bio-Inspired Computation,* vol. 8, no. 4, pp. 205-214, 2016.

[28] G. Wang, L. Guo, H. Wang, H. Duan, L. Liu, and J. Li, "Incorporating mutation scheme into krill herd algorithm for global numerical optimization," *Neural Computing and Applications,* vol. 24, no. 3-4, pp. 853-871, 2014.

[29] G.-G. Wang, A. H. Gandomi, and A. H. Alavi, "Stud krill herd algorithm," *Neurocomputing,* vol. 128, pp. 363-370, 2014.

[30] G.-G. Wang, L. Guo, A. H. Gandomi, G.-S. Hao, and H. Wang, "Chaotic krill herd algorithm," *Information Sciences,* vol. 274, pp. 17-34, 2014.

[31] P. Ngatchou, A. Zarei, and A. El-Sharkawi, "Pareto multi objective optimization." pp. 84-91.

[32] H. Li, M. Dong, K. Ota, and M. Guo, "Pricing and repurchasing for big data processing in multi-clouds," *IEEE Transactions on Emerging Topics in Computing,* vol. 4, no. 2, pp. 266-277, 2016.

[33] S. Karthikeyan, P. Asokan, S. Nickolas, and T. Page, "A hybrid discrete firefly algorithm for solving multi-objective flexible job shop scheduling problems," *International Journal of Bio-Inspired Computation,* vol. 7, no. 6, pp. 386-401, 2015.

[34] S. S. Reddy, B. Panigrahi, S. Debchoudhury, R. Kundu, and R. Mukherjee, "Short-term hydro-thermal scheduling using CMA-ES with directed target to best perturbation scheme," *International Journal of Bio-Inspired Computation,* vol. 7, no. 3, pp. 195-208, 2015.

[35] K. Ota, M. Dong, S. Chang, and H. Zhu, "MMCD: cooperative downloading for highway VANETs," *IEEE Transactions on Emerging Topics in Computing,* vol. 3, no. 1, pp. 34-43, 2015.

[36] B. S. Rao, and K. Vaisakh, "Multi-objective adaptive clonal selection algorithm for solving optimal power flow problem with load uncertainty," *International Journal of Bio-Inspired Computation,* vol. 8, no. 2, pp. 67, 2016.

[37] P. Stodola, and J. Mazal, "Applying the ant colony optimisation algorithm to the capacitated multi-depot vehicle routing problem," *International Journal of Bio-Inspired Computation,* vol. 8, no. 4, pp. 228-233, 2016.

[38] M. Dong, K. Ota, and A. Liu, "RFL: achieving energy efficient privacy preserving in wireless sensor networks," *IEEE Transactions on Emerging Topics in Computing,* 2017.

[39] S. Pronk, I. Pouya, M. Lundborg, G. Rotskoff, B. Wesen, P. M. Kasson, and E. Lindahl, "Molecular simulation workflows as parallel algorithms: the execution engine of Copernicus, a distributed high-performance computing platform," *Journal of Chemical Theory and Computation,* vol. 11, no. 6, pp. 2600-2608, 2015.

[40] M. Zhou, and J. Liu, "A Two-Phase Multiobjective Evolutionary Algorithm for Enhancing the Robustness of Scale-Free Networks Against Multiple Malicious Attacks," *IEEE Transactions on Cybernetics*, 2016.

[41] S. N. Adya, and I. L. Markov, "Fixed-outline floorplanning: enabling hierarchical design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems,* vol. 11, no. 6, pp. 1120-1135, 2003.

[42] R. Liu, S. Dong, X. Hong, and Y. Kajitani, "Fixed-outline floorplanning with constraints through instance augmentation." pp. 1883-1886.

[43] S. Chen, and T. Yoshimura, "Fixed-Outline Floorplanning: Block-Position Enumeration and a New Method for Calculating Area Costs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 27, no. 5, pp. 858-871, 2008.

[44] T.-C. Chen, and Y.-W. Chang, "Modern floorplanning based on B*-tree and fast simulated annealing," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems,* vol. 25, no. 4, pp. 637-650, 2006.

Gai-Ge Wang (M'2015) is an associate professor in Jiangsu Normal University, China. His research interests are swarm intelligence, evolutionary computation, and big data optimization. He has published more than 80 papers. He just proposed four bio-inspired algorithms (monarch butterfly optimization, earthworm optimization algorithm, elephant herding optimization, moth search).

Xingjuan Cai is an Associate Professor in the School of Computer Science and Technology, Taiyuan University of Science and Technology, China. Her current research interest is bio-inspired algorithms and cloud computing. She has published over 20 peer reviewed journal papers, 10 peer reviewed full conference papers in computational intelligence.

Zhihua Cui received the PhD degree in System Engineering from Xi'an Jiaotong University, China. He is a Professor of Computer Science and Technology, Taiyuan University of Science and Technology, China. He is the director of Complex System and Computational Intelligence Laboratory. His research interest includes computational intelligence, big data, cloud computing and related research topics.

Geyong Min is the Chair and Director of High Performance Computing and Networking (HPCN) Research Group at the University of Exeter, UK. He received the PhD degree in Computing Science from the University of Glasgow, UK, in 2003. His main research interests include Next-Generation Internet, Analytical Modelling, Cloud Computing, and Big Data.

Jinjun Chen received the PhD degree in IT from Swinburne University of Technology, Australia. He is a Professor at Swinburne University of Technology, Australia. He is Deputy Director of Swinburne Data Science Research Institute and Director of Swinburne Big Data Lab. His research interests include Big Data, Data Science, Cloud Computing, Scalability, Security and Privacy, and related various research topics. He is a senior member of the IEEE.