IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 14, 2018, accepted September 14, 2018, date of publication September 18, 2018, date of current version October 12, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2870969

Design of an Optimized Architecture for Manned and Unmanned Combat System-of-Systems: Formulation and Coevolutionary Optimization

ZHE SHU, WEIPING WANG, AND RUI WANG, (Member, IEEE)
College of Systems Engineering, National University of Defense Technology, Changsha 410073, China

Corresponding author: Zhe Shu (shuzhe@nudt.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61403404 and in part by the Distinguished Natural Science Foundation of Hunan Province under Grant 2017JJ1001.

ABSTRACT With the rapid advancement of unmanned combat systems and the maturation of unmanned swarm technology, integrating unmanned combat systems into existing combat system of systems has become a hot topic. However, the current research at the organizational level is basically nonexistent. Therefore, we propose a methodology for designing an optimized architecture that is a significant product at this level to fill this technical gap. The design process consists of two steps: architecture modeling and coevolutionary optimization. The architecture model is composed of three basic elements, which are mission, equipment, and the command and control structure. The selection of the best scheme from possible solutions is a multiobjective optimization problem, and its computational complexity increases rapidly with the increase in the scale of combat system of systems. To overcome this difficulty, we introduce a decomposition-based evolutionary algorithm NSGA-III and improve it by using preference vectors to enhance the algorithm's local search capability. Finally, we conduct the comparative experiments on benchmark test suites and design an operation case to demonstrate the advance and effectiveness of the enhanced algorithm. The proposed architecture modeling and optimization method can achieve a set of nondominated solutions and provide auxiliary decision-making information to help commanders make decisions and arrangements.

INDEX TERMS Architecture modeling, coevolutionary optimization algorithm, combat system-of-systems, unmanned combat system.

I. INTRODUCTION

In information warfare, various unmanned combat systems (UCSs) have been widely used on the battlefield. From land robots to unmanned submarines at sea and drones in the air, UCSs take advantage of their strong mobility, low cost and good concealment performance to carry out dull, dirty, and dangerous missions and achieve the target of ''noncontact, zero casualties'' [1]. The United States is the first to carry out research on unmanned system-related technologies and is in the absolute leading position in the field of UCS military applications. After years of research and demonstration, the U.S. military has selected unmanned swarm operation as an important research direction. Under the unified leadership of the Department of Defense (DoD), the Defense Advanced Research Projects Agency (DARPA), the Strategic Competence Office (SCO), the Air Force and Navy have conducted many research studies and demonstration work and initiated several projects. There is no doubt that unmanned swarms will be the main force of the future battlefield [2]. Considering the highly dynamic and uncertain operational environment and humanitarian issues in combat, the swarm global judgment and emergency response capability based on programmatic planning cannot match the capabilities of manned combat systems (MCSs). Therefore, the integration of unmanned swarms into existing combat system-of-systems will be the main form on the future battlefield [3].

As mentioned in [4], the use of UCSs can be divided into 4 levels based on the hierarchical operation. In Figure 1, the four levels are the organizational, brigade, user, and device levels. Currently, the main academic interest lies in the bottom three levels. At the brigade level, research is mainly conducted on swarm operations, including cooperative task

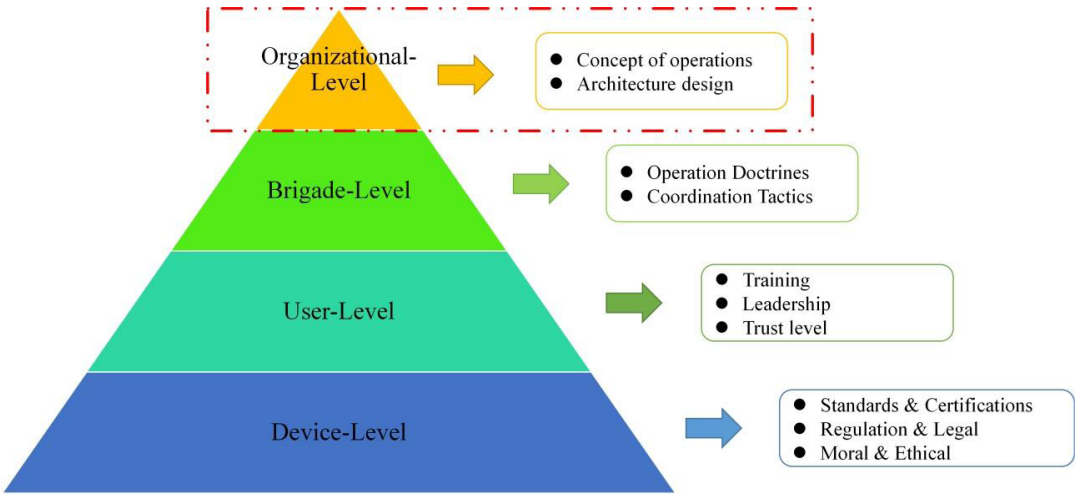

FIGURE 1. Unmanned combat systems in hierarchical operation level.

assignment [5]–[8], collaborative path planning [9]–[11], and collaborative team control [12]–[14]. The user and device levels focus on the fields of platform autonomous control, link communications, human-computer interaction, and human-machine intelligence integration [15]–[17]. However, on the organizational level, a combat system-of-systems (CSoS) containing an unmanned swarm has not received widespread and sufficient attention. From the perspective of system-of-systems (SoS) engineering, it is necessary to design the SoS architecture at the initial stage of construction to ensure it develops in the right direction. Therefore, we believe now is a good time to introduce architecting technology into CSoS containing unmanned swarms.

As an outstanding representative of system-of-systems engineering, the U.S. DoD has performed quite admirable work on SoS architectures. They not only have dealt with large-scale defense projects but also integrated military systems as SoS architectures [18]. The SoS architecture is the carrier of system-of-systems capabilities by integrating all the capabilities from different component systems to accomplish the overall mission goals. Obviously, this definition is based on capability, and the purpose of architecture is to determine which systems gather to form a capable system-of-systems [19]. As mentioned in [20] and [21], the missions drive the architecting process towards the ideal, capable SoS architecture. Therefore, the optimized architecture designed in this paper requires providing a set of capabilities for a specific military mission and simultaneously taking the characteristics of unmanned swarms into account.

In this article, we propose a trinity architecture model, namely, mission, equipment and C2 structure, to describe CSoS through analysis of the capability views. Furthermore, we define an architecture optimization problem, that is, how to design the equipment and C2 relationship of the CSoS for a specific operational mission, so that the CSoS has the best combat performance, lowest operational cost, and minimum collaborative workload for the C2 structure. For such a multi-objective optimization problem with more than two objectives, the efficiency of dominance-based approaches decreases significantly [22]. Thus, a coevolutionary algorithm is presented to solve this optimization problem, and a scenario case is designed to demonstrate the feasibility and effectiveness of the proposed algorithm.

The rest of the paper is organized as follows: in the next section, a literature review will summarize the two aspects of SoS architecture and multiobjective evolutionary algorithms (MOEAs). Section 3 establishes the architecture model according to the characteristics of the CSoS and formulates a multiobjective optimization model. In Section 4, an improved NSGA-III is proposed. Section 5 designs an operational scenario to verify the feasibility of the formulation and the effectiveness of the optimization. Finally, concluding and future work is presented in Section 6.

II. LITERATURE REVIEW

As mentioned in Section 1, the architecting problem includes two parts: architecture modeling and architect optimization. In this section, we will conduct a literature review from both the modeling methods and the optimization algorithms for architecting. The first part mainly introduces three modeling methods for architecting, and analyzes three methods from the perspective of architecture optimization needs (data requirements). For the multi-objective optimization problem studied in this paper, the traditional Pareto-dominance-based approaches have performance deficiencies. In response to these shortcomings, the second part mainly summarizes the improvement measures in three aspects.

A. MODELING METHODS FOR CSoS ARCHITECTING

There are many modeling methods widely applied in SoS architecting, especially for military applications [23], [24]. In this study, we summarize three kinds of CSoS architecture modeling methods according to different formulations.

1) PATTERNS FOR SoS ARCHITECTING

The SoS architecture carries a large amount of information about component systems, including the characteristics of component systems and the interactive relationship between them. A pattern of SoS architecture is regarded as the framework adopted by the DoD Architecture Framework (DoDAF) [25], MOD Architecture Framework (MoDAF) [26], and NATO Architecture Framework (NAF) [27]. It has been designed by top-level SoSE engineers and optimized through multiple views [28], [29]. Additionally, because multiple views have been created to describe the SoS architecture, this approach requires a large quantity of support data. For example, the operation views (OV) describe the operation nodes, combat mission, or operation activities, as well as the information that must be exchanged to complete the mission. Also, there are needed a lot of data to describe the type of information exchange and the frequency of information interactions.

2) AGENT-BASED SoS ARCHITECTING

SoS are becoming increasingly complex as component systems continue to grow in multiple divergent paths consisting of multidistributed, independent, capable and self-organized entities [30]. Agent-based SoS architecting is well accepted as a popular approach that treats component systems as agents. As an outstanding representative, the Agent-based model (ABM) became widespread in the 1990s, and ABM is a collection of abstracted and computational individuals referred to as agents. Thus, one can simulate the interactive behavior and self-determination of intelligent agents [31], [32]. Particularly, the ABM is more popular in the military/defense field because each intelligent individual (commander, soldier, or even a robot with a high level of intelligence) can be modeled as an agent, and the model can truly demonstrate the operation of SoS by defining and regulating the interactions between agents. Agents have their own intelligence when accepting activities based on the whole mission and deal with the activities in each specific circumstance. The autonomy of individuals is the most realistic description of SoS in the objective world. Furthermore, the behavior between agents embodies swarm intelligence and can generate global emergent behavior [32]. Similarly, agent-based architecting requires more data than the first approach to support the modeling of the state and behavior of each agent.

3) ANALYTIC MODELS FOR SoS ARCHITECTING

Analytic models have been widely adopted in defense and military operational research methodologies since as early as the 1970s [33]. SoS architecting becomes a combinatorial problem when all kinds of systems are collected into a joint warfighting system-of-systems, and feasible solutions must be provided by analytic computation [19]. As a combinatorial problem with multiple objectives, the problem can be solved either as a multiobjective optimization problem (MOP) or as a single-objective optimization problem (SOP) by reformulating multiple objectives into one dimension. Obviously, a SOP solver is much easier than the other methods for generating an optimal solution, and the supporting approaches are similar to a simple weighted summation of all the objectives or complicated evaluation in a fuzzy logic system (FLS) [34]. Although this method can quickly reduce the solution space to produce a solution, it contains many subjective factors (such as the weighting ratio and fuzzy knowledge acquisition) that will subjectively mask other useful information in the solution space. Therefore, the MOP solver is needed to generate a set of nondominated solutions to support operational decision-making.

Here, we pay more attention to the multiobjective optimization approach for CSoS architecting issues. For instance, a multiobjective optimal concept is introduced to CSoS construction by Wolf [35]. Agarwal *et al.* [36] developed a computational intelligence approach by using a genetic and particle swarm algorithm. Konur *et al.* [37] establish an indicative function as a CSoS architecture model and adopted an evolutionary method to generate the approximated Pareto fronts. A heuristic approach was successfully used to generate a Pareto Front collection for CSoS architecting issues [24].

According to the literature review, the first two methods require a large amount of data to support construction and optimization. In contrast, the third method can supplement the model with a large amount of prior knowledge and is more suitable for use in top-level design and planning. In addition, it is a MOP that can be significantly handled by multiobjective evolutionary algorithms (MOEAs). However, when the number of objectives increases, the performance of traditional Pareto-dominance-based MOEAs will degrade obviously. The relevant improvements will be summarized in the following part.

B. IMPROVEMENTS FOR PARETO-DOMINANCE-BASED MOEAS

MOEAs have a history of nearly forty years and have been verified as an efficient method to tackle MOPs. Over the past thirty years, Pareto-dominance-based methods, such as NSGA-II [38] and SPEA2 [39], have become the most popular algorithms applied to MOPs with no more than two objectives. Recently, research has proven that Pareto-dominance-based MOEAs have markedly reduced performance as the number of objectives increased [40] because as the objective space dimension increases, Pareto-dominance-based MOEAs do not provide enough pressure to push populations to the frontiers. The improvement work has mainly proceeded via the following three aspects:

1) MODIFICATION OF DOMINANCE RELATION

Because the Pareto-dominance-based methods lose efficacy in the dominance relationship, the first kind of improvement strategy aims to further enhance the dominance of the algorithms. For example, a grid-based evolutionary algorithm (GrEA) was proposed to take advantage of the grid dominance to push the selective

pressure to the frontier [41]. Similar approaches include: ε-dominance [42], θ-dominance [43], L-optimality [44], and so on.

2) REPLACEMENT OF THE DOMINANCE RELATION

Another promising improvement strategy is to completely replace the original dominant strategy. There are two more popular approaches: the indicator-based and decomposition based. The first method hopes to achieve a dominant choice of population by providing new indicators, and it is called an indicator-based evolutionary algorithm (IBEA) [45]. Furthermore, hypervolume (HV) is introduced into the indicator-based method with the merits of the MOEA algorithm and is proposed as a new method, namely, HypE [46]. The second approach is to find a way to decompose the MOP into single-objective subproblems and optimize them at the same time. This kind of method is called a multiobjective evolutionary algorithm based on decomposition (MOEA/D) [47] and has several variants such as NSGA-III [48] and MOLSD [49].

3) LOCAL SEARCH ADJUSTMENT

When the solution space is too large and nondominated solution points cannot be compared, a preference-based local search mechanism is proposed. Kim *et al.* [50] specified the preference as a prior information in front of the global search, and Jaimes *et al.* [51] dealt with the preference information interactively. The preference-inspired coevolutionary algorithm using goal vectors (PICEA-g) [52] utilizes the preference to produce approximation sets for a posteriori decision-making to push the candidate solutions in the optimal direction.

In general, the three categories of MOEAs each have their own advantages. Among them, Deb *et al.* [38] found that NSGA-III performs better than other MOEAs on the DTLZ benchmark with 2 to 15 objectives [47]. Although the NSGA-III promotes the diversity of candidate solutions by applying uniformly distributed reference points, it is still unsatisfactory in terms of the convergence effect [53]. The PICEA approach has been proven to have better performance than the other five state-of-the-art MOEAs for the HV, which is the most important indicator of convergence [52]. In addition, decision makers will provide corresponding preference information as posterior knowledge to assist decision making in different combat environments. Therefore, we chose to combine NSGA-III with the preference-inspired coevolutionary mechanism and propose an improved NSGA-III to solve the MOP in CSoS architecting.

III. FORMULATION FOR CSoS ARCHITECTURE

A. CSoS ARCHITECTURE MODEL

It is necessary to have a definite understanding of capability, which is the basis of the CSoS architecture model. The U.S. DoD is convinced that capability is the ability to achieve a desired effect by using resources, i.e., doctrine, organization, training, materiel, leadership, education, personnel, and facilities (DOTMLPF), under specified standards and conditions [54]. RAND defines capability as the ability to accomplish a set of tasks and achieve the desired results through a set of means or sets of behaviors under the given standards and conditions [55]. From the perspective of capability generation, a capable CSoS usually chooses manned and unmanned systems with different capabilities according to the allocated mission and stipulates the C2 structure between the systems. Obviously, capability is the bridge that forms a direct mapping relationship between the missions (activities list) and systems. Therefore, we established the architecture model by defining the following elements.

1) CAPABILITY

CSoS is capable when it is provided with n capabilities by component systems. Let the capabilities be indexed by i, such that $i \in I = \{1, 2, \cdots, n\}$.

2) MISSION (ACTIVITIES LIST)

A specific mission can be refined into a list of activities that are the smallest mission units, and the activities can be indexed by k, such that $k \in K = \{1, 2, \cdots, l\}$.

The purpose of an architecture is to clearly define the relationship between the mission and capability; therefore, we set up an adjacent matrix of activity-capability, $A = \{a_{ik}\}_{n\times l}$, where a_{ik} is an indicator variable to show whether activity i requires capability k.

$$a_{ik} = \begin{cases} 1 & \textit{if activity } i \textit{ requires capabilty } k, \\ 0 & \textit{otherwise} \end{cases} \tag{1}$$

For each activity i, there is at least one capability to continue the mission, which means $\forall i \in I, \sum_{k\in K} a_{ik} \geq 1$.

3) SYSTEMS (MANNED AND UNMANNED)

There are m_1 manned systems and m_2 unmanned systems providing capabilities to accomplish the mission. Manned systems have a higher level of intelligence and C2 but are fixed in CSoS. Let them be indexed by j_1, such that $j_1 \in J_1 = \{1, 2, \cdots, m_1\}$. Unmanned systems are in a limited level of C2 to prevent inhumane belligerence and have more open architecture interfaces, which can selectively join or exit CSoS.

Let unmanned systems be indexed by j_2, such that $j_2 \in J_2 = \{m_1 + 1, m_1 + 2, \cdots, m_1 + m_2\}$ and $J = J_1 + J_2$. The differentiated classification of systems will involve different effects in the measurement and calculation of operational costs and collaborative workloads.

The selected systems are required to provide relevant capabilities for a specific mission, and thus, we set up the system-capability matrix $B = \{b_{jk}\}_{m\times l}$ and activity-system matrix $X = \{x_{ij}\}_{n\times m}$ as follows:

$$b_{jk} = \begin{cases} 1 & \textit{if system } j \textit{ provides capability } k, \\ 0 & \textit{otherwise} \end{cases} \tag{2}$$

$$x_{ij} = \begin{cases} 1 & \text{if activity } i \text{ selects system } j, \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

where each system j can provide more than one kind of capability, and each activity i will choose more than one system, such that $\forall j \in J, \sum_{k\in K} b_{jk} \geq 1$ and $\forall i \in I, \sum_{j\in J} x_{ij} \geq 1$.

Further, the CSoS must be equipped with enough systems to provide the capabilities to complete the mission. Here, we make an assumption that the performance of different capabilities can be summed up linearly, as introduced in [56].

4) C2 STRUCTURE

When unmanned swarms are introduced into an existing CSoS, the changes that occur in the C2 structure are the most significant, including the swarms' ad hoc network, the collaboration between swarms and manned systems, and the open interfaces for swarms. We assume that all the MCSs are controlled by v_1 C2 units and all the UCSs consist of v_2 swarms, where C2 units and swarms are indexed by u_1 and u_2, respectively, such that $u_1 \in U_1 = \{1, 2, \cdots, v_1\}$ and $u_2 \in U_2 = \{v_1 + 1, v_1 + 2, \cdots, v_1{+}v_2\}$. Meanwhile, each swarm is assigned to a C2 unit, so all the C2 units in the CSoS can be indexed by u and $u \in U = U_1 + U_2$. The integration of swarms into the CSoS operation is divided into three steps as follows:

First, we allocate all the manned combat systems to different C2 units.

$$y_{j_1u_1} = \begin{cases} 1 & \text{if MCS } j_1 \text{ is assigned to C2 unit } u_1, \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

$Y = \left\{y_{j_1u_1}\right\}_{m_1\times v_1}$ presents the adjacent matrix for the MCS- C2 unit connection.

Then, unmanned combat systems are allocated to swarms and further assigned to C2 units.

$$z_{j_2u_2} = \begin{cases} 1 & \text{if UCS } j_2 \text{ is choosen by swarm } u_2, \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $Z = \left\{z_{j_2u_2}\right\}_{m_2\times v_2}$ presents the adjacent matrix for the UCS-swarm connection. Since a swarm corresponds to a C2 unit, the adjacent matrix Z also represents the relationship between the C2 units and UCSs.

Finally, the C2 units are assigned to different activities:

$$c_{iu} = \begin{cases} 1 & \text{if C2 unit } u \text{ is assigned to activity } i, \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

The premise of the judgment condition in (6) satisfies $x_{ij} = 1, y_{ju_1} = 1, j \in J_1$ or $x_{ij} = 1, z_{ju_2} = 1, j \in J_2$, namely, the conditions under which a C2 unit participates in activity i ensures that the systems controlled by the C2 unit, u, are selected by activity i. Therefore, Equation (6) can be transformed into:

$$c_{iu} = \begin{cases} 1 & \text{if there exit system } j, \text{ such that } x_{ij} = 1, \\ & y_{ju_1} = 1, \quad j \in J_1 \text{ or } x_{ij} = 1,\ z_{ju_2} = 1,\ j \in J_2 \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

$$c_{iu} = x_{ij} \cdot \left(y_{j_1u_1} + z_{j_2u_2}\right) \tag{8}$$

Let $C = \{c_{iu}\}_{n\times v} = X \cdot (Y + Z)$ be the activity-C2 unit adjacent matrix. Further, we can define the collaboration in activity i as:

$$Co_i = \sum\nolimits_{u,w=1}^{v} c_{iu}c_{iw} = \sum\nolimits_{u,w=1}^{v} \min\left(c_{iu}, c_{iw}\right) \tag{9}$$

B. OBJECTIVE FUNCTIONS FOR SoS ARCHITECTURE

In architecting of the component systems collection, there are some indices that need to be noted. Acheson *et al.* [57] use performance, affordability, robustness, modularity, security, etc. as the key performance parameters/attributes (KPP/KPA) of the architecture [57]. As a pioneer in SoS architecting optimization, the DoD believed that agility, performance, and cost could effectively characterize SoS performance [58]. Further, Konur and Dagli noted that performance, completion time, and total cost are the objectives of the architecture [56]. It can be concluded that performance and cost are the main two attributes for CSoS architecting. Meanwhile, the collaborative workload in cooperation is another important indicator for integrating unmanned swarms into a CSoS. Therefore, a CSoS architecting problem is defined to construct a capable and stable CSoS with maximum operation performance, minimum total cost and minimum total collaborative workload.

1) TOTAL PERFORMANCE

Let per_{jk} be the performance level that a selected system j provides in capability, and let $Perf$ be the $m \times l$ matrix of per_{jk} values, namely $Per = \left\{per_{jk}\right\}_{m\times l}$. The CSoS architecting performance includes the overall performance of various systems displayed for all the capabilities. Before formulating the performance function, there are two assumptions. We assume that the total performance is the sum of all kinds of capability performances, and the performance of a specific capability is the sum of that capability's performance provided by the component systems through collaboration. The total performance can be calculated by two parts. One is the inherent system performance, and the other is the performance increase generated by multiple systems collaborating to complete the same activity. Thus, we can formulate the performance function as follows.

$$\begin{aligned} Per(X, Y, Z) &= \left(1 + Co_i \Big/ \sum_{i\in I} Co_i\right) X \cdot Per \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{k=1}^{l}\left(1 + Co_i \Big/ \sum_{i\in I} Co_i\right)\left(x_{ij} \cdot per_{jk}\right) \end{aligned} \tag{10}$$

Specifically, collaboration involves both Y and Z matrices. Therefore, the performance function contains three matrix variables, X, Y, and Z.

2) TOTAL COST

The total cost of a military SoS can be divided into three parts, namely, the operating cost for providing capabilities, the collaboration cost for multiple systems operating in the same activity, and the communication cost between component systems. The operating cost is relevant to component systems involved in CSoS, and the communication cost depends on the organization structure. Therefore, they can be expressed as:

$$\begin{aligned} &Cost(X, Y, Z) \\ &= X \cdot Cost^{op} + (Y + Z) \cdot Cost^{com} + Co \cdot Cost^{col} \\ &= \sum_{i=1}^{n}\sum_{j=1}^{m} x_{ij} \cdot cost_j^{op} + \sum_{u=1}^{v}\sum_{j=1}^{m} \left(y_{uj} + z_{j_2u_2}\right) \cdot cost_j^{com} \\ &\quad + \sum_{i=1}^{n} Co_i \cdot cost_i^{col} \end{aligned} \tag{11}$$

where $cost_j^{op}$ and $cost_j^{com}$ represent the operating and communication costs of system j, respectively, and $cost_i^{col}$ is the cost for the collaborative operation in activity i. Further, let $Cost^{op}$, $Cost^{com}$, and $Cost^{col}$ be the vector sets representing each cost variable.

3) COLLABORATIVE WORKLOAD

The CSoS is an organic whole in which each component unit needs to accomplish activities by collaborating with other units, and this process will produce a collaborative workload. For each C2 unit u, the collaborative workload can be divided into two categories, namely, the internal collaborative workload (ICW) and external collaborative workload (ECW). We characterize the ICW as the number of systems simultaneously controlled by the C2 unit p, such that:

$$ICW_u = \sum_{j_1=1}^{m_1} y_{j_1u_1} + \sum_{j_2=m_1+1}^{m_1+m_2} z_{j_2u_2} \tag{12}$$

Furthermore, the ECW represents the collaborative relationship between C2 unit p and the others, such that:

$$ECW_u = \sum_{i=1}^{n} c_{iu}c_{iw} = \sum_{i=1}^{n} \min\left(c_{iu}, c_{iw}\right) \tag{13}$$

Therefore, the total collaborative workload can be calculated as:

$$CW\,(Y, Z) = \sum_{u=1}^{v} \left(W^I \cdot ICW_u + W^E \cdot ECW_u\right) \tag{14}$$

where W^I and W^E indicate the weights of ICW and ECW, respectively.

C. OPTIMIZATION PROBLEM OF CSoS ARCHITECTURE

Here, we can define the optimization problem of CSoS architecture (CSoSAOP) as the maximum performance, minimum cost and coordination workload with the matrix variables.

$$\begin{aligned} CSoSAOP: &\max Perf\,(X, Y, Z) \\ &\min Cost(X, Y, Z) \\ &\min CW\,(Y, Z) \end{aligned} \tag{15}$$

Meanwhile, we should take the following constraints into account.

$$\forall i \in I, \quad \sum_{j\in J} x_{ij} \geq 1 \tag{16}$$

$$\forall i \in I, \quad \sum_{k\in K} a_{ik} \geq 1 \tag{17}$$

$$\forall j \in J, \quad \sum_{k\in K} b_{jk} \geq 1 \tag{18}$$

$$\forall i \in I, \quad \sum_{j\in J}\sum_{k\in K} b_{jk} \cdot x_{ij} \geq \sum_{k\in K} a_{ik} \tag{19}$$

$$\forall u \in U, \quad \sum_{j\in J} \left(y_{j_1u} + z_{j_2u}\right) \geq 1 \tag{20}$$

$$\forall u, w \in U, \quad \min\left(c_{iu}, c_{iw}\right) \geq x_{ij} \cdot \left(y_{j_1u} + z_{j_2u}\right) \tag{21}$$

Accordingly, the constraints (16)-(18) and (20) are to ensure that all the adjacent matrices are meaningful. Constraint (19) guarantees that a CSoS can provide enough capabilities for each activity by selecting adequate MCSs and UCSs. Constraint (21) limits the collaboration between different C2 units when they are allocated to the same activity.

Obviously, CSoSAOP is a nonlinear integer multiobjective optimization problem that can be solved by MOEAs. The number of CSoSAOP is three, and the Pareto-dominance based MOEAs fail to provide sufficient selection pressure in the optimal direction. To solve this problem, Section 4 will introduce an improved NSGA-III to solve the multi-objective optimization problem (MOP) in CSoS architecting.

IV. AN ENHANCED NSGA-III WITH A PREFERENCE-INSPIRED CO-EVOLUTIONARY MECHANISM

As reviewed in Section 2, the traditional Pareto-dominance based MOEAs, such as NSGA-II and SPEA2, lose their power to have the desired Pareto sets, and three improvements have been made. NSGA-III, an outstanding improved MOEA based on decomposition, has been demonstrated to perform much better than other state-of-the-art MOEAs on well-known benchmarks [48]. However, NSGA-III has been improved in terms of diversity, and there is still room to enhance the algorithm convergence. Therefore, we introduce a preference-inspired coevolutionary mechanism to enhance the local search of NSGA-III, namely, a preference vector-oriented NSGA-III (P- NSGA-III). In this section, the framework of the improved algorithm will be first proposed, and then, we will implement the enhanced NSGA-III and adjust the crossover and mutation operators according to the constraint in CSoSAOP.

A. FRAMEWORK OF P-NSGA-III

The P-NSGA-III algorithm is based on the NSGA-III framework and borrows some ideas, such as ideal point generation, genetic operation and adaptive normalization. Meanwhile, NSGA-III has great performance on diversity,

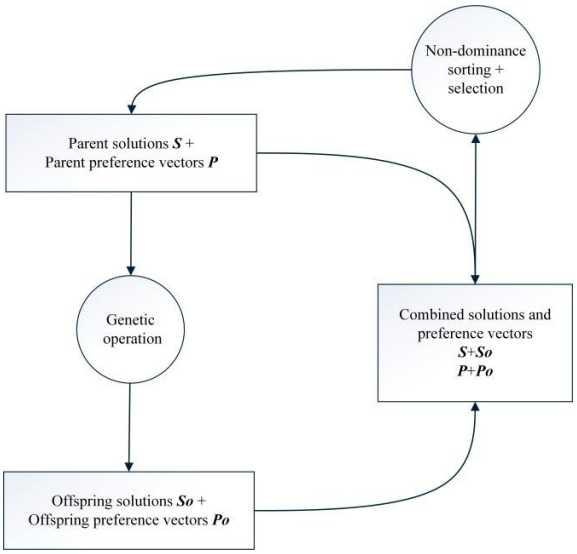

FIGURE 2. An elitist framework of P-NSGA-III.

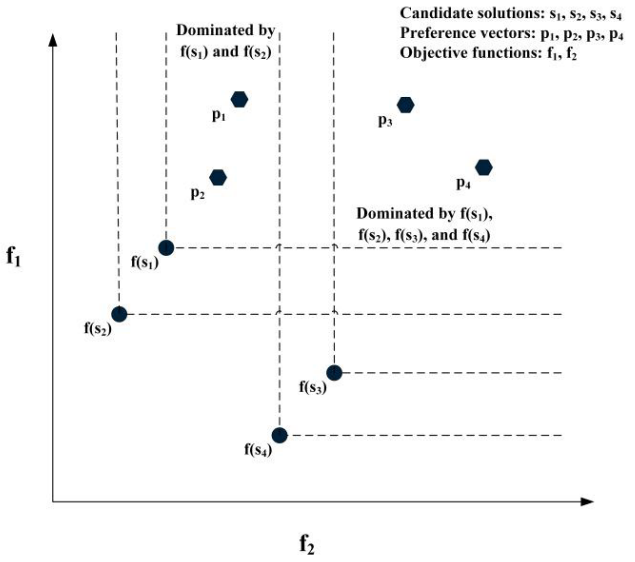

FIGURE 3. Simple biobjective minimization problem with four candidate solutions and four preference vectors.

and the preference vectors can effectively improve the capability of local search. In Figure 2, the elitist framework of P-NSGA-III is shown. The parent solutions S and preference vectors P have fixed populations, N and Np. Both of them are evolved for Maxgen generations. In each generation t, the parent solution set $S(t)$ is subjected to genetic operations to generate N offspring solution sets $So(t)$. Meanwhile, the parent preference vectors $P(t)$ also produce the offspring $Po(t)$ according to genetic operations. Then, parents and offspring are pooled together $S(t)$ and $So(t)$, $P(t)$ and $Po(t)$, respectively. The combined preference vectors provide important nondominance information to help truncate the combined solution set, and the nondominance mechanism will be introduced next. Then, the combined solutions are divided into several different nondominance levels ($F1$, $F2$, etc.) based on the preference vectors supporting nondominance information. Assuming that the last acceptable nondominant level is the l level, the solutions at the $l+1$ level and beyond are discarded, and the solutions in $S(t)+So(t)\backslash Fl$, the parent solutions in generation t and the selected solutions from level 1 to level $l-1$, are chosen as the solution in $S(t+1)$. The remaining individuals in $S(t+1)$ need to be selected from the Fl. The basis for selection is to make the solutions have ideal diversity in the objectives space. Similarly, the preference vectors $P(t+1)$ will be generated in an analogous process except for the different Pareto dominance.

The preference vector mechanism borrows an idea from the PICEA-g algorithm [52], which is demonstrated on a biobjective minimization problem in Figure 3. In the objectives space, some individuals can determine the dominance relationship with others, such as s_1 and s_2, s_3 and s_4, but there are some individuals who cannot directly determine the dominance relationship with other individuals, such as s_2 and s_4. Here, we introduce the preference vectors as supporting information to provide selective pressure toward the Pareto front. The fitness functions of the candidate solutions and preference vectors are formulated as follows:

$$F_s = 0 + \sum_{p \in P \cup P_0} \frac{1}{n_p} \tag{22}$$

$$F_p = \frac{1}{1+\alpha} \tag{23}$$

$$\alpha = \begin{cases} 1, & n_p = 0 \\ \dfrac{n_p - 1}{2N-1}, & otherwise \end{cases} \tag{24}$$

where n_p denotes the number of solutions that dominate the preference vector p_i.

In Figure 3, we define the "f" notation is an objective function. There are four functions of candidate solutions, namely $f(s_1), f(s_2), f(s_3)$ and $f(s_4)$, displayed on a biobjective minimization problem space. Also the four preference vectors are randomly generated in Figure 3. Further, preference vectors p_1 and p_2 are dominated by solutions s_1 and s_2, and preference vectors p_3 and p_4 are dominated by solutions s_1, s_2, s_3 and s_4. Further, we can conclude that $n_{p_1} = n_{p_2} = 2$ and $n_{p_3} = n_{p_4} = 4$. Therefore, the solution fitness can be calculated as:

$$F_{s_1} = F_{s_2} = \frac{1}{n_{p_1}} + \frac{1}{n_{p_2}} + \frac{1}{n_{p_3}} + \frac{1}{n_{p_4}} = \frac{3}{2} \tag{25}$$

$$F_{s_3} = F_{s_4} = \frac{1}{n_{p_3}} + \frac{1}{n_{p_4}} = \frac{1}{2} \tag{26}$$

Based on the fitness, s_1 and s_2 are considered the best solutions, which will be selected for the next generation. However, obviously s_1 is dominated by s_2. Compared with s_1, although s_4 has a lower fitness, it is non-dominated with s_2. Therefore, s_4 and s_2 are desired to be kept in the population set. In order to do that, the classic Pareto-dominance

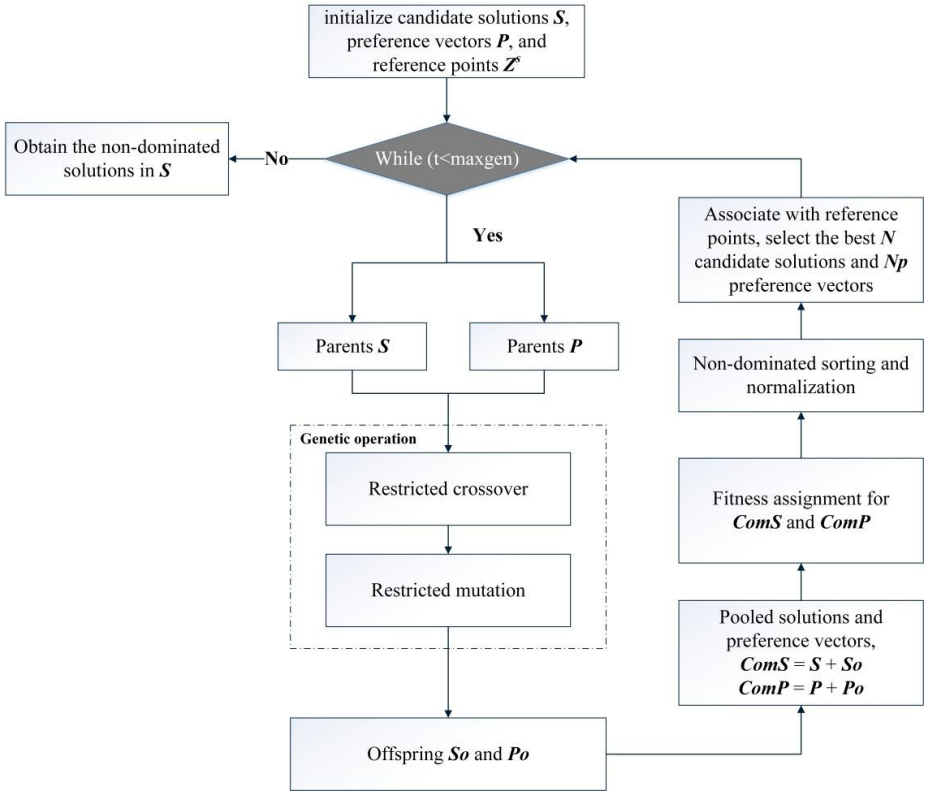

FIGURE 4. The flow chart of P-NSGA-III.

relation is incorporated. After calculating fitness values using (22)–(24), we next identify all the non-dominated solutions in $S \cup So$. If the number of non-dominated solutions does not exceed the population size, then we assign the maximum fitness to all the non-dominated solutions. However, if more than N non-dominated solutions are found, we then disregard the dominated solutions prior to applying truncation selection (implicitly, their fitness is set to zero). Based on fitness, the best N non-dominated solutions are selected to constitute the new parent $S(t+1)$. In the example of Figure 3, $F_{s_1} = F_{s_3} = 0$, $F_{s_2} = 3/2$, and $F_{s_4} = 1/2$. The preference vector fitness can be calculated in a similar way.

Since NSGA-III still uses Pareto-dominance-based sorting method, the performance achieved in the more than two conflict objectives optimization problem is not satisfactory. As illustrated above, the preference vector mechanism provides a new idea of dominating sorting, and enhances the local search capability of the original algorithm. It makes the obtained sorting results more accurate, and further improves subsequent hierarchical operations based on the reference points. Eventually the performance of NSGA-III is improved.

In the course of combat, we are more hopeful that we can achieve higher combat performance. We can use the specific preferences in these applications as an a priori guide to the generation of preference vectors so that the optimization process is more realistic. Considering the preference vectors, the flow chart of P-NSGA-III is more specific in Figure 4.

In Figure 4, we further refine the algorithm flow and specifically describe how to use preference vectors to enhance the nondominated sorting performance of NSGA-III. In Algorithm 1, the computational process of the algorithm in the t generation is presented through pseudocode. As indicated in original NSGA-III, we predefine a set of reference points to ensure the diversity in filtered solutions. According to [48], we can predefine structured reference points $\boldsymbol{Z^s}$ and supplied aspiration points $\boldsymbol{Z^a}$.

In the pseudocode, $P(t)$ and $Po(t)$ are respectively the selected preference vectors and the offspring preference vectors in generation t. The preference generator currently generates random preference vectors before variation operators haven't been implemented for the preference vectors. More specifically, preference vectors are randomly generated as objective vectors directly in objective space, within bounds defined by the vectors of ideal and anti-ideal performance. $Q(t)$ is a new population of combined solutions, which has l levels to maintain the diversity of the solutions. The function "fitness" of the solutions and preference vectors are formulated in Equations (22) and (23), respectively. The preference vectors coevolve with the candidate solutions in the same generation.

Before the coevolutionary optimization of CSoSAOP, the candidate solution encoding procedure is necessary. In this research, there are three matrix variables X, Y, and Z. We convert the matrix variables into vectors by extracting

Algorithm 1 P-NSGA-III Procedure in Generation t

Input: selected solutions, $S(t)$, selected preference vectors, $P(t)$, structured reference points Z^s or supplied aspiration points Z^a
Parameter: N indicates the scale of initial candidate solutions, M indicates the size of the objectives, N_p indicates the number of goal vectors, H denotes the number of reference points, l indicates the number of nondominance levels, and pc and pm represent the probability of crossover and mutation, respectively.
Output: Promoted candidate solutions, $S(t+1)$, promoted preference vectors, $P(t+1)$.

i=1, $Q(t) = \emptyset$% $Q(t)$ is the hierarchy set of populations.
[$So(t)$, $Po(t)$] = **crossover_mutation** ($S(t)$, $P(t)$, pc, pm)
f(S) = **objective_function** (S)
ComS = **merge** ($S(t)$, $So(t)$)
Comf = **merge** (f($S(t)$), f($So(t)$))
ComP = **merge** ($P(t)$, $Po(t)$)
[FitComS, FitComP] = **fitness** (Comf, ComP)
[Fit_1, Fit_2, ...] = **nondominated_sorting** (FitComS, FitComP)
repeat
  $Q(t) = Q(t)$ U Fit_i, $i = i + 1$
until $|Q(t)| \geq$ N, $Fit_i = Fit_l$
if $|Q(t)|$ = N **then**
  $S(t+1) = Q(t)$, break
else
  $Q(t) = \bigcup_{j=1}^{l-1} Fit_j$
  $K = N$ - $|Q(t)|$ % choose the other points from Fit_l
  (f^n, Z^r) = **normalize** ($Q(t)$, Z^s, Z^a) [48] % normalize objectives and create reference set
  $S(t+1)$ = **associate_filtrate** ($Q(t)$, Z^r) % associate individuals in $Q(t)$ with a reference point and filtrate N offspring solutions.
  $P(t+1)$ = **associate_filtrate** ($Q(t)$, Z^r) % truncate N_p offspring preference vectors
end if

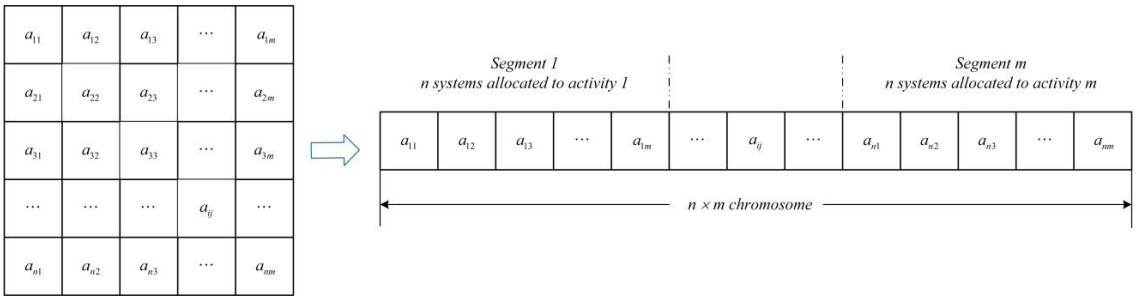

FIGURE 5. Schematic of solution encoding for matrix variables.

matrix row vectors, as illustrated in Figure 5. For a CSoSAOP solution, the entire length of the chromosomes is $n \times m + m_1 \times q_1 + m_2 \times q_2$.

B. SOLUTION ENCODING

The gene on the chromosomes indicates the adjacency of matrix variables, which is a 0-1 value. Actually, this simplifies our crossover and mutation process introduced in the next subsection.

C. RESTRICTED CROSSOVER AND MUTATION

The algorithm uses two classical genetic operator codings: simulated binary crossover (SBX) and polynomial mutation (PM) [38]. Because the three matrix variables all represent adjacency relationships, the chromosomes of different variables all have the 0-1 encoding form. When we perform cross-mutation operations, we do not need to process the three matrices independently. Instead, we can combine the three matrix codes as a whole to perform cross and mutation operations, as long as the three matrices after cross and mutation satisfy the constraints. This greatly reduces the complexity and workload of the crossover and mutation. It is not necessary to pay attention to the crossover and mutation process. We just need to check whether the new solutions meet the constraints (16)-(21) at the end of the crossover and mutation process.

V. COMPUTATIONAL EXPERIMENTS

In this section, we will verify the excellence and effectiveness of P-NSGA-III through two tests, namely benchmark test and operation case study. The benchmark test problems are chosen from the standard MOEAs' test problem suite, and the proposed algorithm will be compared with other five state-of-the-art MOEAs on the standard MOEA's performance metric, hypervolume. The operation case study is abstracted from actual requirements of CSoS architecture design. We will compare P-NSGA-III algorithm with the original NSGA-III in this case.

A. BENCHMARK TEST

1) TEST PROBLEMS

The test problems are chosen from the MaF [59] benchmark test suite, which is proposed as the benchmark test suite applied in IEEE CEC'2017. The MaF benchmark test suite includes fifteen test functions, which are a good abstraction and integration of different multiobjective optimization problems in the real world. According to [59], the properties of the fifteen are list in Table 1.

TABLE 1. The properties of fifteen MaF benchmark test problems.

Problem	Properties
MaF1	Linear
MaF2	Concave
MaF3	Convex, multimodal
MaF4	Concave, multimodal
MaF5	Convex, biased
MaF6	Convex, degenerate
MaF7	Mixed, disconnected, multimodal
MaF8	Linear, degenerate
MaF9	Linear, degenerate
MaF10	Mixed, biased
MaF11	Convex, disconnected, nonseparable
MaF12	Concave, nonseparable, biased deceptive
MaF13	Concave, unimodal, nonseparable, degenerate
MaF14	Linear, partially separable, large scale
MaF15	Convex, partially separable, large scale

Considering the space limitations and the properties of real case, we select six representative problems as the benchmark test suite for this experiment. They are MaF1, MaF6, MaF8, MaF10, MaF11, and MaF15.

2) COMPARATIVE ALGORITHMS AND PERFORMANCE METRICS

As mentioned in Section 2, there are three methods to improve the performance of traditional Pareto-dominance-based MOEAs on many-objective optimization problems. Li *et al.* [49] have summarized five categories of thirteen MOEAs, which are all state-of-the-art in each field. As shown in Table 2, we select five representatives from the five categories as the comparison group.

There are two widely accepted performance metrics, hypervolume (HV) [46] and inverted generational distance (IGD) [47], measuring both convergence and diversity of MOEAs. Among them, IGD needs to know the true PF data to calculate the distances of the each point in true PF to the points in its nearest approximation front. While, the follow-up case is abstracted from the real problem to be solved, and its true PF is unknown. To ensure the continuity of the experiment, we choose HV as the performance metric. The HV metric selects a reference point in the objective space, which is dominated by all optimal candidate solutions. By measuring the size of domination space, we can calculate the HV value. Given a reference point, a lager HV value indicates better performance.

TABLE 2. The representative MOEAs selected from five categories.

MOEAs	Categories	Properties
MOEA/D	C1	Decomposition-based approaches
HyPE	C2	Indicator-based approaches
NSGA-Ⅲ	C1,C3	Selection criterion modified approaches
GrEA	C4	Dominance relation modified approaches
PICEA-g	C5	Preference-based approaches.

3) PARAMETER SETTINGS

In this part, the general parameter settings are introduced first, followed by specific parameter settings for each individual algorithm.

- For each MaF test problem, there are three parameters to set. Assume that D is the number of decision variables, M is the number of objectives, and K is the position parameter. In our tests, M is taken as 3, 5, 7, 10 or 15, D is obtained by $D = M + K - 1$, where K is set to 10 for MaF1, MaF6 and 20 for the other problems.
- For the experimental settings, we set N is the size of population to 100. The comparative results are obtained through 31 independent experiments for each algorithm, which makes the results more statistically significant [52].
- Parameters for Crossover and Mutation Operator: The simulated binary crossover is used as the crossover operator in all the mentioned algorithms. In detail, the distribution index is set to $\eta_c = 15$ in HypE, GrEA, and PICEA-g, $\eta_c = 30$ in P-NSGA-III, MOEA/D, and NSGA-III. The crossover probability $p_c = 1.0$ is set for all algorithms. For the polynomial mutation, the distribution index is set to $\eta_m = 20$ and the mutation probability is set to $p_m = 1/nvar$, where $nvar$ represents the number of decision variables.
- For each comparative MOEAs, we set their specific parameters according to the original papers or related literatures, which are shown in Table 3. It is worth noting that the best parameter settings of GrEA vary according to different test problems. However, we use the same set of parameters for all problem instances, which is a little different from the best one. As discussed in [60], this difference should not affect the final evaluation results.

TABLE 3. The specific parameter settings for comparative MOEAs.

MOEAs	Parameters
P-NSGA-III	The number of reference points $H \approx N = 100$, the number of preference vectors: $Np = 100$
MOEA/D	The penalty parameter θ of the PBI function: 5, neighborhood size: $N = 10$
HypE	The number of sampling points: 10,000
NSGA-III	The number of reference points $H \approx N = 100$
GrEA	The grid division $div = 15$ for 3 objectives, $div = 10$ for 5 objectives, $div = 8$ for 7 objectives, $div = 14$ for 10 objectives, and $div = 12$ for 15 objectives
PICEA-g	The number of preferences: $NGoal = 100$

4) RESULTS ANALYSIS

To further verify the effectiveness of P-NSGA-III, we make a series of comparison with five MOEAs on MaF test suite. Meanwhile, in order to ensure the validity of the statistical results, we set up the mean and standard deviation of HV on 31 independent experiments to evaluate the performance of these algorithms.

Table 4 lists HV values for the five algorithms on MaF test suite, marking the best results for each set of experiments in blue. We conducted Wilcoxon rank sum test in the six competing evolutionary algorithms to determine whether P-NSGA-III is statistically superior to the others. For each test instance, the best performance is highlighted in blue. The results of tests are written in Table 4 and marked with symbols $+$, $\approx$, or $-$, indicating that P-NSGA-III performed significantly superior to, approximately equal to, and significantly inferior to the competing algorithms. The last rows of Table 4 summarize the alignments on each test suite in which P-NSGA-III is significantly better than, equal to or worse than the others.

Table 4 shows that P-NSGA-III performs well on all the instances with the HV indicator, and has obvious advantages compared to other algorithms. For all the test suites, P-NSGA-III achieved the best in 19 of 30, which was much better than other five state-of-art MOEAs. Compared with MOEA/D, P-NSGA-III performs well on almost all the instances, except for four test problem. Compared with HypE and GrEA, P-NSGA-III significantly surpassed MaF1, MaF10 and MaF11, by both getting more than 10 '-' signals, meaning they were much better than other MOEAs. For NSGA-III and PICEA-g, P-NSGA-III borrows a lot from evolutionary mechanism of these two algorithms, and has more '$\approx$' signals relatively.

It is noted that P-NSGA-III obtains 16 '-' signals of 30 and 3 best results of 5 by comparing all three-objective MaF test problems. We can indicate that P-NSGA-III will perform better on operation case study in the next part.

B. OPERATION CASE STUDY

Area reconnaissance and denial (ARD) is a classical operation scenario involving air, ground and underwater combat systems. In the past ten years, as a large number of UCSs have entered the battlefield and unmanned swarm technologies have become increasingly mature, the CSoS in ARD needs to be further adjusted. In this section, a novel ARD is designed as a case study for CSoSAOP, and the general parameter settings are listed in detail. After that, we quantitatively analyze the solution methods proposed for CSoSAOP through a set of numerical studies.

1) CASE DESCRIPTION

Here, we design a novel ARD application scenario for the case demo, which draws on experience [61]. Particularly, we choose seven capabilities, six kinds of systems (no more than 40), and six C2 units containing two swarms for the ARD mission, as shown in Tables 5 and 7. It is worth noting that we can generate a matrix based on Table I and build a matrix using Table 3. Furthermore, when we take the calculation of capability and cost into consideration, we should make some assumptions as follows.

- As shown in Table 7, we assume that the performance of each system enabling different activities can be divided into levels of 1 to 5 by expert knowledge. According to the definition of capability in Section 3, each capability has the same importance; hence, the total performance is calculated by summing all the individual performances.
- In Table 7, we divide all the systems into two categories, i.e., manned and unmanned combat systems, and give the number of limitations for each kind of system. Based on the assumption of interaction between all systems, we believe that the internal and external loads have the same importance, that is, $W^I = W^E = 0.5$.
- There are three types of cost for systems operating in ARD missions, which are assumed to be between ¥10K and ¥100K RMB ($K = 10^3$). Here, satellites, airships, and panzers are man-controlled and can be categorized as manned combat systems; UAV, UGV, and loitering missiles are swarm-controlled and can regarded as unmanned combat systems. We assume that all the enabled systems exert their capabilities in the mission to generate a sum of cost, shown as a detailed list in Table 6. Finally, we assume that all the systems use Link16 as their communication network, so the communication interface costs are the same and equal ¥ 10K for each. The collaborative cost for manned and unmanned combat systems is ¥ 20K and ¥ 10K, respectively.

According to the above table, we can initialize the candidate solutions that contain N matrix variables subject to constraints (16)-(21). In addition, then, we can obtain an objective space based on objective functions (10), (11), and (14) and define a set of reference points. In this objective space, the Np preference vectors are randomly generated based on operational preference information, which is the better combat performance in this case.

2) GENERAL PARAMETER SETTINGS

In this application, we initialize the size of the candidate solutions and preference vectors both to 100 and set the maximum generation to 100. In the crossover operator, the crossover

TABLE 4. The statistical results (mean and standard deviation) of the HV values obtained by P-NSGA-III, MOEA/D, HypE, NSGA-III, GREA, and PICEA-G on MaF test suite.

Problem	M	D	PNSGAIII	MOEAD	HypE	NSGAIII	GrEA	PICEAg
MaF1	3	12	2.5587e-1 (5.28e-2)	2.6340e-1 (1.23e-3) +	1.8794e-1 (1.46e-2) -	2.6928e-1 (1.48e-3) +	8.6384e-2 (1.30e-2) -	2.5440e-1 (3.36e-3) ≈
	5	14	1.2170e-2 (6.22e-3)	5.0280e-3 (2.25e-3) -	2.7612e-3 (2.74e-4) -	5.8988e-3 (1.09e-3) -	3.0424e-3 (6.40e-4) -	1.1846e-2 (9.04e-4) ≈
	7	16	2.8640e-4 (2.77e-5)	2.4378e-5 (1.04e-5) -	5.6628e-5 (9.94e-6) -	1.5333e-4 (2.56e-5) -	5.8532e-5 (1.03e-5) -	2.7340e-4 (2.80e-5) ≈
	10	19	4.7630e-7 (1.06e-7)	2.5367e-8 (9.71e-9) -	1.3949e-7 (6.93e-8) -	4.4746e-7 (7.83e-8) ≈	1.6989e-7 (3.79e-8) -	3.7931e-7 (1.11e-7) -
	15	24	4.7992e-12 (1.40e-12)	1.3639e-13 (3.30e-14) -	1.4250e-12 (1.04e-12) -	2.9461e-12 (7.25e-13) -	6.3856e-12 (1.54e-12) +	3.5364e-12 (5.36e-13) -
MaF6	3	12	1.3541e-1 (6.05e-2)	9.2717e-2 (4.44e-2) -	1.2671e-1 (3.40e-3) ≈	1.2781e-1 (6.66e-4) ≈	8.6644e-2 (6.27e-3) -	1.1686e-1 (5.89e-3) -
	5	14	8.2629e-3 (4.04e-4)	3.2977e-3 (3.25e-3) -	8.5470e-3 (6.46e-4) ≈	8.6597e-3 (1.53e-4) ≈	7.9312e-3 (4.68e-4) ≈	8.3358e-3 (2.68e-4) ≈
	7	16	1.9907e-4 (4.37e-5)	1.1932e-4 (9.31e-5) -	1.9109e-4 (2.26e-5) ≈	1.5992e-4 (7.94e-5) ≈	1.9719e-4 (5.35e-6) ≈	1.9841e-4 (3.18e-6) ≈
	10	19	5.4521e-8 (1.81e-8)	1.8713e-8 (2.64e-8) -	5.4129e-8 (4.77e-9) ≈	2.6719e-8 (1.35e-8) -	1.0811e-8 (2.41e-8) -	5.0169e-8 (6.07e-10) ≈
	15	24	2.3489e-17 (4.02e-17)	8.4611e-17 (5.13e-18) +	5.3941e-17 (3.64e-17) ≈	1.0176e-17 (2.55e-17) ≈	2.6473e-17 (6.51e-17) ≈	1.5696e-17 (3.49e-17) ≈
MaF8	3	2	5.9263e-1 (4.80e-1)	2.4269e-1 (3.54e-1) -	5.8950e-1 (4.59e-1) ≈	4.4793e-1 (5.14e-1) ≈	3.4329e-1 (3.01e-1) -	3.8430e-1 (3.67e-1) -
	5	2	1.4309e+0 (9.90e-1)	7.7606e-1 (8.14e-1) -	8.6242e-1 (6.06e-1) -	1.3484e+0 (1.29e+0) ≈	7.6373e-1 (5.03e-1) -	8.7880e-1 (7.74e-1) -
	7	2	2.5958e+0 (1.60e+0)	1.8284e+0 (1.70e+0) ≈	2.3375e+0 (1.81e+0) ≈	3.1149e+0 (2.60e+0) ≈	1.6916e+0 (1.53e+0) ≈	1.3255e+0 (1.52e+0) ≈
	10	2	3.1770e+0 (2.68e+0)	2.5291e+0 (2.80e+0) ≈	3.3121e+0 (4.41e+0) ≈	5.2279e+0 (4.49e+0) ≈	2.9631e+0 (2.33e+0) ≈	3.7579e+0 (3.08e+0) ≈
	15	2	1.6936e+1 (1.02e+1)	1.5679e+0 (2.74e+0) -	2.7047e+0 (5.38e+0) -	1.6514e+1 (1.05e+1) ≈	7.8119e+0 (7.38e+0) -	6.1254e+0 (8.05e+0) -
MaF10	3	22	2.4692e+1 (1.25e+0)	2.2145e+1 (1.78e+0) -	1.6481e+1 (2.30e+0) -	2.3228e+1 (9.90e-1) -	1.9276e+1 (2.29e+0) -	2.4741e+1 (1.40e+0) ≈
	5	24	2.4226e+3 (1.45e+2)	2.0261e+3 (2.11e+2) -	2.0810e+3 (3.44e+2) -	1.9872e+3 (6.10e+1) -	2.3288e+3 (2.24e+2) ≈	2.3487e+3 (1.17e+2) ≈
	7	26	4.5391e+5 (2.96e+4)	4.0100e+5 (2.69e+4) -	4.1238e+5 (3.38e+4) -	3.4998e+5 (2.09e+4) -	4.3155e+5 (4.91e+4) ≈	4.5376e+5 (4.74e+4) ≈
	10	29	3.2124e+9 (1.91e+8)	2.3150e+9 (2.92e+8) -	3.1371e+9 (3.76e+8) ≈	2.4436e+9 (1.66e+8) -	3.1267e+9 (3.60e+8) ≈	2.2053e+9 (2.36e+8) -
	15	34	5.0080e+16 (1.72e+15)	3.1059e+16 (2.47e+15) -	4.4818e+16 (5.74e+15) ≈	4.5879e+16 (5.31e+15) ≈	4.4527e+16 (9.61e+15) ≈	4.9499e+16 (2.95e+15) ≈
MaF11	3	22	5.7178e+1 (6.84e-1)	4.8459e+1 (2.82e+0) -	5.6036e+1 (6.30e-1) -	5.6100e+1 (6.20e-1) -	3.6878e+1 (1.75e+0) -	5.7077e+1 (3.57e-1) ≈
	5	24	5.8241e+3 (5.68e+1)	4.6342e+3 (2.08e+2) -	5.7030e+3 (8.43e+1) -	5.5998e+3 (6.64e+1) -	2.6741e+3 (5.08e+2) -	5.8695e+3 (7.77e+1) ≈
	7	26	1.1916e+6 (1.82e+4)	8.9314e+5 (3.64e+4) -	1.1794e+6 (8.65e+3) -	1.1385e+6 (2.46e+4) -	4.3336e+5 (1.05e+5) -	1.1970e+6 (1.45e+4) ≈
	10	29	9.1200e+9 (1.43e+8)	7.1338e+9 (6.67e+8) -	8.9636e+9 (1.62e+8) -	8.4892e+9 (5.15e+8) -	3.9948e+9 (9.55e+8) -	9.0852e+9 (1.26e+8) ≈
	15	34	1.6710e+17 (4.03e+15)	1.1528e+17 (7.07e+15) -	1.5501e+17 (1.50e+16) ≈	1.3735e+17 (1.17e+16) -	6.0299e+16 (2.41e+16) -	1.6703e+17 (5.41e+15) ≈
MaF15	3	26	9.6624e-1 (8.72e-2)	4.5715e-1 (5.60e-2) +	4.9203e-2 (5.65e-2) ≈	5.4621e-2 (6.48e-2) ≈	1.6770e-1 (1.01e-1) ≈	1.3651e-1 (8.57e-2) ≈
	5	30	4.4234e-2 (2.95e-3)	3.9669e-2 (2.87e-2) ≈	1.5520e-3 (1.51e-3) -	1.7246e-2 (3.72e-3) -	3.1421e-3 (4.79e-3) -	6.0634e-4 (9.08e-4) -
	7	36	6.5302e-6 (8.60e-6)	1.1832e-5 (2.30e-5) +	5.2137e-7 (1.01e-6) -	6.1742e-6 (3.18e-6) ≈	5.3136e-6 (6.95e-5) ≈	2.9584e-6 (3.98e-6) ≈
	10	43	3.0254e-6 (3.75e-6)	1.1746e-6 (1.33e-6) ≈	2.4937e-7 (6.45e-7) -	9.1724e-7 (7.41e-7) -	8.4099e-7 (1.62e-6) -	1.7376e-7 (2.64e-7) -
	15	58	5.7813e-13 (8.77e-13)	4.9089e-13 (6.36e-13) ≈	1.4982e-14 (3.81e-14) -	3.7813e-14 (8.77e-14) -	2.8814e-15 (8.54e-15) -	1.6028e-14 (2.57e-14) -
+/-/≈				4/21/5	0/18/12	1/16/13	1/18/11	0/10/20

The best results are highlighted in blue.
+ means that this MOEA shows significantly better performance than P-NSGA-III in the comparison.
- means that this MOEA shows significantly worse performance than P-NSGA-III in the comparison.
≈ means that there is no significant difference between the compared results.

TABLE 5. Activities and capabilities required in ARD mission.

Activity	i	Capability	k	Capability needs
warning surveillance	1	Fast search	1	20
		Area patrol	2	
		Continuous trail	3	
Location	2	Precise location	4	15
Denial	3	High-speed reach	5	30
		Survivor removal	6	
		Suppressive fire	7	

TABLE 6. Cost in ARD mission.

System	Category	Cost(CNY)
Satellite	Operating cost	¥20K
Airship		¥50K
Unmanned aerial vehicle		¥60K
Unmanned ground vehicle		¥45K
Panzer		¥80K
Loitering missile		¥100K
All	Communication cost	¥10K
Unmanned combat systems	Collaborative cost	¥10K
Manned combat systems	Collaborative cost	¥20K

probability $p_c = 1$ is set for the algorithm, and the distribution index is set to $\eta_c = 15$. For the polynomial mutation, the distribution index is set to $\eta_m = 20$, and the mutation probability is set to $p_m = 1/nvar$, where nvar represents the number of decision variables.

3) FIGURE RESULTS ANALYSIS

The enhanced and original NSGA-III are both applied in the ARD case study, and the comparison of the last generation is demonstrated in the three-dimensional and two-dimensional representations in Figure 6.

Figure 6 provides the three-dimensional and two-dimensional comparisons of the last generation. The black circles represent the Pareto front generated by the original NSGA-III, and the red stars represent the P-NSGA-III optimal results. Based on the experimental results, we can draw the following conclusions.

- There are 28 and 29 optimal solutions obtained by the enhanced and original NSGA-III, respectively, which are much smaller than the 100 population size.

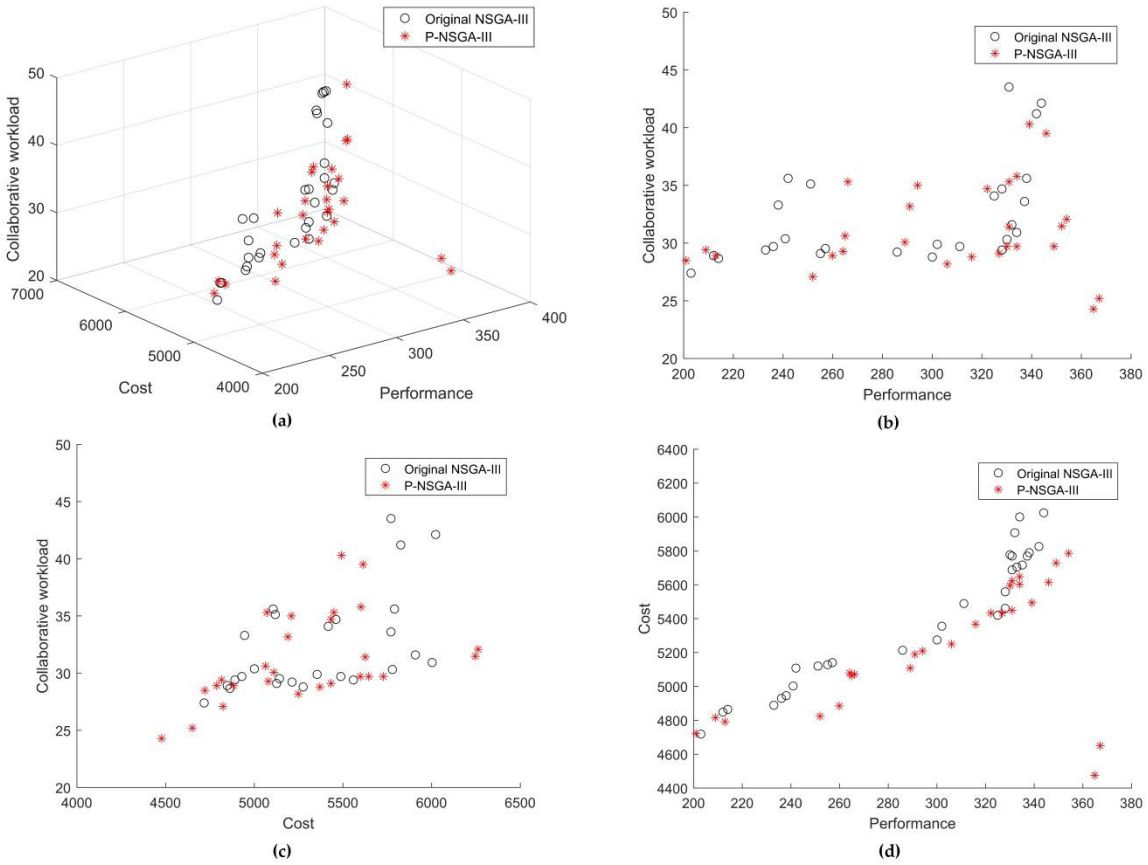

FIGURE 6. Comparison of the last generation generated by two algorithms: (a) The three-dimensional Pareto front representation; (b)-(d) The projection of the three-dimensional Pareto front onto two dimensional planes.

Because the genetic operation is subject to constraints (16)-(21) and the integer requirements of the variables, the space of executable solutions is limited, and part of the optimal solutions are repeated. In Figure 6, duplicate solutions are covered reducing the number of final results.

- In Figure 6, we can clearly observe that the P-NSGA-III solutions have higher performance, lower cost and collaborative workload than those solutions obtained by the original algorithm showing that P-NSGA-III performs better in dealing with ARD problems.

Figure 6 shows that the performance of optimal solutions is approximately linear and positive with cost, which is in line with the objective world law. The higher the input cost is, the greater the gain is.

4) ANALYTICAL RESULTS ANALYSIS

In this part, we choose the standard MOEAs' performance metric, to evaluate the Pareto fronts obtained by the two algorithms. To ensure the statistical significance of the experimental results, the two algorithms are taken 31 independent runs on the ARD problem.

The ARD problem is abstracted from the actual engineering requirements and hasn't been solved yet, so we cannot obtain the true Pareto Front of ARD problem. As analyzed in the previous, IGD needs to know the true PF data to calculate the distances. Therefore, we select HV as the performance metric to assess the two algorithms.

HV is taken to calculate the volume of the objective space between the obtained solution set and a specified reference point, which is strict Pareto-compliant to make the consequence rather fair. HV works as follows. Let A be the final non-dominated points set, which is obtained by a MOEA algorithm in the objective space, and $r = (r_1, r_2, \ldots, r_l)$ be a reference point. The HV value of final nondominated points set A is the volume of the region dominated by A with regard to r, which is formulated as follow:

$$HV(A, r) = volume\left(\bigcup_{f \in A} [f_1, r_1] \times [f_2, r_2] \times \ldots \times [f_m, r_m]\right) \tag{27}$$

For a given reference point r, the larger value indicates the better performance. To facilitate the comparison of the two sets of PFs, we generate a common reference point for the calculation of HV metric. Hence, we select a given reference point r in the objective space, which is weekly dominated by all optimal solutions of the combined PF set.

TABLE 7. Systems applied in ARD mission.

Category	System	Number	Activity	C2 unit	Capability	levels
Manned	Satellite	≤2	1, 2	1,2	2, 4	1
	Airship	≤5	1	1,3	2, 3, 4	2
	Panzer	≤4	3	3	5, 6, 7	2
Unmanned	Unmanned aerial vehicle	≤15	1, 2	4	1, 2, 3, 4	4
	Loitering missile	≤5	1, 3	4,5	1, 2, 3, 5, 6, 7	5
	Unmanned ground vehicle	≤10	1, 2	6	1, 2, 3, 4	3

Thus, we can obtain two sets of HV values. The HV values can be taken as the statistics to assess the superiority of P-NSGA-III and the original NSGA-III through a nonparametric statistical test, as shown in Algorithm 2.

Algorithm 2 Generation and Nonparametric Test for HV

Input: Pareto fronts obtained by P-NSGA-III, $PF = \{pf_1, pf_2, \ldots, pf_{31}\}$, Pareto fronts obtained by NSGA-III, $PF' = \{pf'_1, pf'_2, \ldots, pf'_{31}\}$, Given reference point, $\boldsymbol{r}$.
Output: HV of P-NSGA-III Pareto fronts, $HV = \{hv_1, hv_2, \ldots, hv_{31}\}$, HV of NSGA-III Pareto fronts, $HV' = \{hv'_1, hv'_2, \ldots, hv'_{31}\}$, Wilcoxon's ranksum test results, $(\boldsymbol{g}, \boldsymbol{h})$.

1. ComPF = **merge** (***PF***, ***PF'***)
2. $\boldsymbol{r} = [\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3]$
3. **for each** Pareto front $\boldsymbol{pf_i}$ or $\boldsymbol{pf'_i} \in$ ComPF**do**
4. **if** $\boldsymbol{r}$ is dominated by $\boldsymbol{pf_i}$ or $\boldsymbol{pf_i}$' **then**
5. $\boldsymbol{hv_i}$ = hypervolume ($\boldsymbol{pf_i}$, $[\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3]$, 10000)
6. $\boldsymbol{hv'_i}$ = hypervolume ($\boldsymbol{pf_i}$', $[\boldsymbol{r}_1, \boldsymbol{r}_2, \boldsymbol{r}_3]$, 10000)
7. **else**
8. **end**
9. $\boldsymbol{HV} = \{\boldsymbol{hv_1}, \boldsymbol{hv_2}, \ldots, \boldsymbol{hv_{31}}\}$
10. $\boldsymbol{HV'} = \{\boldsymbol{hv'_1}, \boldsymbol{hv'_2}, \ldots, \boldsymbol{hv'_{31}}\}$
11. **end**
12. $(\boldsymbol{g}, \boldsymbol{h})$ = **ranksum** (***HV***, ***HV'***, 0.05)

The performance of the original NSGA-III is statistically compared with P-NSGA-III by a nonparametric statistical test called Wilcoxon's ranksum test for independent samples with significance level of 0.05. The comparative result is shown in Table 8.

TABLE 8. Nonparemetric statistical test result for the two comparative algorithms.

Parameter	Value
g	1.8267e-04
h	1

In Table 8, g is the probability of significance that two population samples are consistent. If g is close to 0, the inconsistency is more obvious. Also, the result $h = 1$ indicates a rejection of the null hypothesis [62]. Therefore, the results of Wilcoxon's ranksum test show that P-NSGA-III performs better than the original algorithm in ARD application based on the HV metric.

In this section, the benchmark test is first demonstrated the advance of P-NSGA-III by comparing the other five MOEAs. And then, an operation case study shows the feasibility of CSoSAOP modeling and the effectiveness of enhanced NSGA-III. This method can effectively solve CSoSAOPs and provide useful auxiliary decision-making information to help commanders make decisions and arrangements.

VI. CONCLUSIONS AND FUTURE WORK

The methodology of CSoS architecting optimization fills the technical gap in unmanned and manned integrated operations and is an important attempt at system-of-systems engineering in the unmanned battlefield. This paper is divided into two parts including architectural modeling and coevolutionary optimization to complete the design of an optimal CSoS architecture. First, we built a CSoS architecture model from three perspectives, mission, equipment, and C2 structure combined with the characteristics of unmanned combat, and proposed the corresponding architectural optimization problem, namely, CSoSAOP. For CSoSAOP, we introduced a decomposition-based evolutionary algorithm NSGA-III and improved it by using preference vectors to enhance the local search capability. Finally, we designed and demonstrated an area reconnaissance and denial (ARD) application scenario for the case demo, which verified the effectiveness of the modeling and optimization method.

On one hand, we must acknowledge that many assumptions and simplifications have been made during the design and demonstration of the case. There is still a large gap between the case and practical application. This is also a direction for us to further research by creating more practical cases in the next step. On the other hand, this is still an offline planning optimization method at present. In the future, we hope to improve the accuracy and timeliness of the calculations by improving the evolutionary algorithm and to create an online decision support system for the construction of CSoS architecting.

REFERENCES

[1] Q. Zhu, R. Zhou, and J. Zhang, "Connectivity maintenance based on multiple relay UAVs selection scheme in cooperative surveillance," *Appl. Sci.*, vol. 7, no. 1, pp. 1–8, Dec. 2016.
[2] P. Garcia-Aunon and A. B. Cruz, "Comparison of heuristic algorithms in discrete search and surveillance tasks using aerial swarms," *Appl. Sci.*, vol. 8, no. 5, p. 711, May 2018.
[3] B. Walter, A. Sannier, D. Reiners, and J. Oliver, "UAV swarm control: Calculating digital pheromone fields with the GPU," *J. Defense Model. Simul., Appl., Methodol., Technol.*, vol. 3, no. 3, pp. 167–176, Jul. 2006.

[4] K. A. Demir, H. Cicibas, and N. Arica, "Unmanned aerial vehicle domain: Areas of research," *Defense Sci. J.*, vol. 65, no. 4, pp. 319–329, Jul. 2015.
[5] Y.-F. Liu and A. Zhang, "Cooperative task assignment method of manned/unmanned aerial vehicle formation," *Syst. Eng. Electron.*, vol. 3, p. 32, Mar. 2010.
[6] L. Bo, W. Yuanxun, Z. Yanbo, and C. Daqing, "Cooperative task assignment algorithm of manned/unmanned aerial vehicle in uncertain environment," in *Proc. IEEE 2nd Inf. Technol. Netw. Electron. Autom. Control Conf. (ITNEC)*, Chengdu, China, Dec. 2017, pp. 1119–1123.
[7] T. Shima, S. J. Rasmussen, A. G. Sparks, and K. M. Passino, "Multiple task assignments for cooperating uninhabited aerial vehicles using genetic algorithms," *Comput. Oper. Res.*, vol. 33, no. 11, pp. 3252–3269, Nov. 2006.
[8] W. Xinzeng, C. Linlin, L. Junshan, and Y. Ning, "Based on PSO algorithm multiple task assignments for cooperating UAVs," in *Proc. IEEE Int. Conf. Educ. Inf. Technol.*, Chongqing, China, Sep. 2010, pp. V2-25–V2-28.
[9] K.-H. Kou, J.-Y. Yu, G. Wang, and F.-X. Zhang, "Task assignment and route planning method of cooperative attack for manned/unmanned aerial vehicles," in *Proc. IEEE Int. Conf. Unmanned Syst. (ICUS)*, Beijing, China, Oct. 2017, pp. 168–176.
[10] T. W. McLain and R. W. Beard, "Cooperative path planning for timing-critical missions," in *Proc. Amer. Control Conf.*, Denver, CO, USA, Jun. 2003, pp. 296–301.
[11] Q. Hu, J. Zhao, and L. Han, "Cooperative path planning for intelligent vehicle using unmanned air and ground vehicles," in *Proc. Chin. Intell. Syst. Conf.*, Singapore, 2018, pp. 603–611.
[12] M. R. Napolitano, "Development of formation flight control algorithms using 3 YF-22 flying models AFOSR project report," Dept. Mech. Aerosp. Eng., West Virginia Univ., Morgantown, WV, USA, Tech. Rep. ADA434499, 2005.
[13] G. Campa, M. R. Napolitano, B. Seanor, and M. G. Perhinschi, "Design of control laws for maneuvered formation flight," in *Proc. Amer. Control Conf.*, Boston, MA, USA, Jun./Jul. 2004, pp. 2344–2349.
[14] J. Ghommam, H. Mehrjerdi, M. Saad, and F. Mnif, "Formation path following control of unicycle-type mobile robots," *Robot. Auto. Syst.*, vol. 58, no. 5, pp. 727–736, 2010.
[15] A. Bürkle, F. Segor, and M. Kollmann, "Towards autonomous micro UAV swarms," *J. Intell. Robotic Syst.*, vol. 61, no. 1, pp. 339–353, Jan. 2011.
[16] A. Harris, J. J. Sluss, H. H. Refai, and P. G. LoPresti, "Alignment and tracking of a free-space optical communications link to a UAV," in *Proc. 24th Digit. Avionics Syst. Conf.*, Washington, DC, USA, Oct./Nov. 2005, pp. 1.C.1–1.C.9.
[17] R. M. Taylor, "Human automation integration for supervisory control of UAVs," Defense Sci. Technol. Lab., London, U.K., Tech. Rep. RTO-MP-HFM-136, 2006.
[18] J. K. Bergey *et al.*, "U.S. Army workshop on exploring enterprise, system of systems, system, and software architectures," Softw. Eng. Inst., Carnegie-Mellon Univ. Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-2009-TR-008, 2009.
[19] D. M. Curry and C. H. Dagli, "A computational intelligence approach to system-of-systems architecting incorporating multi-objective optimization," *Procedia Comput. Sci.*, vol. 44, pp. 86–94, Jan. 2015.
[20] J. C. Domerçant and D. N. Mavris, "Measuring the architectural complexity of military systems-of-systems," in *Proc. Aerosp. Conf.*, Big Sky, MT, USA, Mar. 2011, pp. 1–16.
[21] J. A. Smith, J. Harikumar, and B. G. Ruth, "An army-centric system of systems analysis (SOSA) definition," Army Res. Lab., Adelphi, MD, USA, Tech. Rep. ARL-TR-5446, Nov. 2011.
[22] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired co-evolutionary algorithms using weight vectors," *Eur. J. Oper. Res.*, vol. 243, no. 2, pp. 423–441, Jun. 2015.
[23] N. Davendralingam and D. DeLaurentis, "A robust optimization framework to architecting system of systems," *Procedia Comput. Sci.*, vol. 16, pp. 255–264, Jan. 2013.
[24] D. Konur and C. H. Dagli, "Military system of systems architecting with individual system contracts," *Optim. Lett.*, vol. 9, no. 8, pp. 1749–1767, Dec. 2015.
[25] *DoD Architecture Framework Working Group, DoD Architecture Framework Version 1.0*, Dept. Defense, Washington, DC, USA, 2003.
[26] I. Bailey, "Brief introduction to MODAF with v1.2 updates," in *Proc. IET Seminar Enterprise Archit. Framework*, London, U.K., 2008, pp. 1–18.
[27] H. A. H. Handley and R. J. Smillie, "Architecture framework human view: The NATO approach," *Syst. Eng.*, vol. 11, no. 2, pp. 156–164, Jun. 2008.
[28] A. Hunton *et al.*, "Use of DoDAF and M&S for the design requirements and optimization of a GIG-enabled wideband mesh-networking waveform," in *Proc. IEEE Mil. Commun. Conf. (MILCOM)*, Washington, DC, USA, Oct. 2006, pp. 1–7.
[29] C. Kerns, A. Brown, and D. Woodward, "Application of a DoDAF total-ship system architecture in building naval ship operational effectiveness models," in *Proc. ASNE Global Deterrence Defense Symp.*, Bloomington, IN, USA, 2011, pp. 14–16.
[30] X. Xia, J. Wu, C. Liu, and L. Xu, "A model-driven approach for evaluating system of systems," in *Proc. 18th Int. Conf. Eng. Complex Comput. Syst. (ICECCS)*, Singapore, Jul. 2013, pp. 56–64.
[31] E. Bonabeau, "Agent-based modeling: Methods and techniques for simulating human systems," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 3, pp. 7280–7287, May 2002.
[32] D. Helbing, "Agent-based modeling," in *Social Self-Organization*, D. Helbing, Ed. Berlin, Germany: Springer, 2012, pp. 25–70.
[33] J. S. Przemieniecki, *Mathematical Methods in Defense Analyses*. Reston, VA, USA: AIAA, 2000.
[34] L. A. Zadeh, "Soft computing and fuzzy logic," *IEEE Softw.*, vol. 11, no. 6, pp. 48–56, Nov. 1994.
[35] R. A. Wolf, "Multiobjective collaborative optimization of systems of systems," Ph.D. dissertation, Dept. Ocean Eng., Massachusetts Inst. Technol., Cambridge, MA, USA, 2005.
[36] S. Agarwal, L. E. Pape, and C. H. Dagli, "A hybrid genetic algorithm and particle swarm optimization with type-2 fuzzy sets for generating systems of systems architectures," *Procedia Comput. Sci.*, vol. 36, pp. 57–64, Jan. 2014.
[37] D. Konur, H. Farhangi, and C. H. Dagli, "On the flexibility of systems in system of systems architecting," *Procedia Comput. Sci.*, vol. 36, pp. 65–71, Jan. 2014.
[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[39] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," in *Evolutionary Methods for Design, Optimization, and Control With Applications to Industrial Problems*, Athens, Greece: International Center for Numerical Methods in Engineering, 2002, pp. 95–100.
[40] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary many-objective optimization: A short review," in *Proc. IEEE Congr. Evol. Comput., IEEE World Congr. Comput. Intell.*, Hong Kong, Jun. 2008, pp. 2419–2426.
[41] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 721–736, Oct. 2013.
[42] A. G. Hernández-Díaz, L. V. Santana-Quintero, C. A. C. Coello, and J. Molina, "Pareto-adaptive ε-dominance," *Evol. Comput.*, vol. 15, no. 4, pp. 493–517, Dec. 2007.
[43] Y. Yuan, H. Xu, B. Wang, and X. Yao, "A new dominance relation-based evolutionary algorithm for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 1, pp. 16–37, Feb. 2016.
[44] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *IEEE Trans. Syst. Man, Cybern. B, Cybern.*, vol. 38, no. 5, pp. 1402–1412, Oct. 2008.
[45] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *Parallel Problem Solving from Nature—PPSN VIII*. Berlin, Germany: Springer, 2004, pp. 832–842.
[46] J. Bader and E. Zitzler, "Hype: An algorithm for fast hypervolume-based many-objective optimization," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, Mar. 2011.
[47] Q. Zhang and H. Li, "MOEA/D: A multiobjective evolutionary algorithm based on decomposition," *IEEE Trans. Evol. Comput.*, vol. 11, no. 6, pp. 712–731, Dec. 2007.
[48] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.
[49] X. Li and M. Li, "Multiobjective local search algorithm-based decomposition for multiobjective permutation flow shop scheduling problem," *IEEE Trans. Eng. Manag.*, vol. 62, no. 4, pp. 544–557, Nov. 2015.
[50] J.-H. Kim, J.-H. Han, Y.-H. Kim, S.-H. Choi, and E.-S. Kim, "Preference-based solution selection algorithm for evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 1, pp. 20–34, Feb. 2012.

[51] A. López Jaimes, A. A. Montaño, C. A. C. Coello, "Preference incorporation to solve many-objective airfoil design problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, New Orleans, LA, USA, Jun. 2011, pp. 1605–1612.
[52] R. Wang, R. C. Purshouse, and P. J. Fleming, "Preference-inspired coevolutionary algorithms for many-objective optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 4, pp. 474–494, Aug. 2013.
[53] X. Bi and C. Wang, "An improved NSGA-III algorithm based on elimination operator for many-objective optimization," *Memetic Comput.*, vol. 9, no. 4, pp. 361–383, Dec. 2017.
[54] *DoD Architecture Framework Working Group, Department of Defense Architecture Framework Version 2.0*, Dept. Defense, Washington, DC, USA, 2009.
[55] P. K. Davis, R. D. Shaver, and J. Beck, *Portfolio-Analysis Methods for Assessing Capability Options*. Arlington, VA, USA: The Rand Corporation, 2008.
[56] D. Konur, H. Farhangi, and C. H. Dagli, "A multi-objective military system of systems architecting problem with inflexible and flexible systems: Formulation and solution methods," *OR Spectr.*, vol. 38, no. 4, pp. 967–1006, Oct. 2016.
[57] P. Acheson, L. Pape, C. Dagli, N. Kilicay-Ergin, J. Columbi, and K. Haris, "Understanding system of systems development using an agent- based wave model," *Procedia Comput. Sci.*, vol. 12, pp. 21–30, Jan. 2012.
[58] J. M. Kaplan, "A new conceptual framework for net-centric, enterprise-wide, system-of-systems engineering," Center Technol. Nat. Secur. Policy, Defense Technol., Nat. Defense Univ., Washington, DC, USA, Tech. Rep. OBM 0704-0188, 2006.
[59] R. Cheng *et al.*, "A benchmark test suite for evolutionary many-objective optimization," *Complex Intell. Syst.*, vol. 3, no. 1, pp. 67–81, Mar. 2017.
[60] K. Li, R. Wang, T. Zhang, and H. Ishibuchi, "Evolutionary many-objective optimization: A comparative study of the state-of-the-art," *IEEE Access*, vol. 6, no. 1, pp. 26194–26214, May 2017.
[61] S. Tangredi, *Anti-Access Warfare: Countering Anti-Access and Area-Denial Strategies*. Annapolis, MD, USA: Naval Institute Press, 2013.
[62] C. N. Gnanaprakasam and K. Chitra, "Soft-computing based digital filter design to analyze vibration signals of induction motor," in *Proc. Int. Conf. Algorithms, Methodol., Models Appl. Emerg. Technol. (ICAMMAET)*, Chennai, India, Feb. 2017, pp. 1–4.

ZHE SHU received the M.S. degree in management science and engineering from the National University of Defense Technology, Changsha, China, in 2014.

He is currently pursuing the Ph.D. degree with the College of Systems Engineering, National University of Defense Technology. His current research interests include evolutionary computation, multiobjective optimization, system-of-systems architecture modeling, and optimization.

WEIPING WANG received the B.S., M.S., Ph.D. degrees from the National University of Defense Technology (NUDT), Changsha, China.

He is currently a Professor with NUDT. His research interests are system-of-systems engineering and complex system simulation.

RUI WANG received the B.S. degree from the National University of Defense Technology, Changsha, China, in 2008, and the Ph.D. degree from The University of Sheffield, Sheffield, U.K., in 2013.

He is currently a Lecturer with the National University of Defense Technology. His research interests include evolutionary computation, multiobjective optimization, machine learning, and various applications using evolutionary algorithms.