Neurocomputing 598 (2024) 128025

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Determining r- and (r, s)-robustness of multiagent networks based on heuristic algorithm

Jie Jiang [a], Yiming Wu [a,*], Zhaoming Zhang [a], Ning Zheng [a], Wei Meng [b]

[a] School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China
[b] School of Automation, Guangdong University of Technology, Guangzhou 510006, China

ARTICLE INFO

Communicated by L. Ma

Keywords:
Multiagent networks
Directed graphs
Cyber security
Network robustness
Heuristic algorithm

ABSTRACT

The graph properties of r- and (r, s)-robustness are of importance for multiagent networks since these properties ensure consensus among agents even in the presence of a limited number of arbitrarily misbehaving agents, given sufficiently large values for the integers r and s. However, determining the exact solutions of r- and (r, s)-robustness of an arbitrary directed graph has been proven to be computationally complex NP-hard problem. In this paper, we introduce a novel method, named the Determining Robustness based Genetic Algorithm (DRSGA), for approximately calculating the r- and (r, s)-robustness using heuristic algorithm. According to graph theory in mathematical analysis, we first formalize the method for calculating the r- and (r, s)-robustness in directed graphs by utilizing a three-vertex set partition. Then, we transform these methods into a minimization problem of an n-dimensional discrete function and employ DRSGA to obtain an approximate solution. Finally, we validate the efficacy of our algorithm through a series of experiments compared to existing mixed-integer programming algorithms.

1. Introduction

Consensus control, as one of the most fundamental issues in coordinated control of multiagent networks, has received widespread attention from researchers in various fields such as distributed optimization [1], multi-robot collaborative manipulation [2], clock synchronization [3], complex networks [4], and smart grids [5]. Owing to the susceptibility of information shared over unsecured communication channels, the multiagent network is at risk of malicious cyber-attacks. Taking into account the cybersecurity, the resilient consensus problem of multiagent networks have attracted considerable attention recently [6–10]. The main objective of resilient consensus is to achieve a shared state agreement among all regular agents in a network, even in the presence of adversarial agents whose identities remain unknown to the regular agents. Many resilient scalar consensus algorithms [11–13] have been proposed for discrete- and continuous-time multiagent networks in the presence of Byzantine nodes in recent years. And the authors in [14–16] investigated the algorithms for achieving resilient vector consensus in the presence of Byzantine attacks.

An important prerequisite for many existing resilient consensus algorithms (including ARC-P [11], W-MSR [12], SW-MSR [17], DP-MSR [18]) to converge is that the corresponding network communication graph satisfies the properties known as r-robustness and (r, s)-robustness [12]. However, existing literature all assumed the availability of such graph properties without addressing methods for their measurement and determination. Unfortunately, from an algorithmic point of view, the problem of determining the r-robustness and (r, s)-robustness of an arbitrary digraph has been proven to be NP-hard and verifying an arbitrary digraph's r-robustness is coNP-complete, as noted in [12,19]. In other words, finding a polynomial-time algorithm for this problem is impossible unless P = NP. To tackle this challenging problem, several approaches have been proposed. The first known algorithm attempting to determine (r, s)-robustness was presented in [20], called DetermineRobustness, which uses exhaustive search to identify the maximum values of r and s for an arbitrary digraph. This method explores the entire search space, becoming highly inefficient as the number of vertices increases. In addition, the authors in [21] presented another algorithm that reduces the problem to mixed-integer linear programming (MILP). There are also some methods that determine r and s for digraphs with a specific structure in polynomial time, including graph construction methods in [12,22], which increase the graph size given fixed values of r and s; the relationship between lower bounding r and the algebraic connectivity, isoperimetric constant of undirected graphs in [23]; and the proof of functional relationships between r and certain graph properties in [17,24–26]. Different from the methods mentioned above, another approach is to obtain the values of such graph properties by solving for approximate solutions. The authors in [27] attempt to obtain the values of r and s using

* Corresponding author.
E-mail address: ymwu@hdu.edu.cn (Y. Wu).

https://doi.org/10.1016/j.neucom.2024.128025
Received 7 December 2023; Received in revised form 22 April 2024; Accepted 7 June 2024
Available online 11 June 2024

machine learning method for a fast, approximate solution based on a trained model. Besides, in [28], the authors propose a sample-based approximate solution algorithm to approximately test r that can adjust accuracy and running time as needed.

Despite the significant findings in previous studies, determining (r, s)-robustness for a large-scale arbitrary digraph is almost impossible [21]. Furthermore, even when using machine learning method [27] to estimate the (r, s)-robustness value, the approach is also infeasible due to insufficient large-scale samples available for model training. Therefore, finding a practical and efficient algorithm to approximate the values of r and s remains an unresolved challenge in this field.

Meta-Heuristics (MHs) are commonly employed in solving Combinatorial Optimization Problems [29], serving as computational intelligence paradigms that provide a structured approach to designing heuristic algorithms delivering satisfactory solutions efficiently. The main objective of MHs is to uncover the optimal solution by emulating human expertise, experience, or natural processes. In contrast to conventional mathematical techniques, heuristic algorithms emphasize exploration within an approximate solution space to expediently yield superior outcomes. Inspired by this approach, in this paper, we adopt a heuristic algorithm that aims to provide an approximate solution of r- and (r, s)- robustness for a given digraph. However, designing a heuristic algorithm directly based on the definition of r and s is quite challenging. On one hand, this is because the definition of $(r,\ s)$-robust requires consideration of every pair of non-empty sets of vertices, which means that the nodes selected for computation in each iteration may not correspond to the size of the vertices in the original graph. On the other hand, the conditions within the definition of $(r,\ s)$-robust are complex, leading to the issue that not every subset of nodes selected for determining s will necessarily yield a valid s value for reference.

Motivated by the discussion above, this paper introduces a method for calculating the r- and (r, s)-robustness of directed graphs by partitioning three sets of vertices. Then, the problem is transformed into the minimization of an n-dimensional discrete function. Finally, a heuristic algorithm named Determining Robustness based Genetic Algorithm (DRSGA) is proposed to obtain approximate solutions of the function. Compared with the existing results, the contributions of this paper are summarized as follows.

1. In contrast to the previously introduced schemes in [20,21,27,28], we establish a formal solution to determine the r-robustness of digraphs by partitioning them into three vertex sets, and then, we transform this issue into a minimization problem for a discrete function with n-elements.
2. For the (r, s)-robustness of a multi-agent network, we formulate the calculation of this metric as an n-dimensional integer nonlinear programming problem with constraints. Then, we introduce a heuristic algorithm-based solution approach and derive an approximate solution for the stated problem.
3. The proposed method is computationally efficient, rendering it highly suitable for deployment in large-scale multiagent networks.

The remainder of this paper is organized as follows: Section 2 presents preliminaries on graph theory and the problem formulation. Section 3 addresses the partition standard formula for determining r-robustness based on the partition of three vertex sets and the problem's equivalent transformation. In Section 4, we discuss seeking the values of s for which a digraph is $(r,\ s)$-robust for a given r by transforming the problem into a discrete function for solving the minimization problem using the exterior point method. In Section 5, we design a heuristic method for the minimization problem of discrete functions. Simulations are presented in Section 6, and Section 7 concludes the paper.

Notations: Let $\mathbb{R}$, $\mathbb{R}_+$, $\mathbb{Z}$, and $\mathbb{Z}_+$ denote the set of real numbers, positive real numbers, integers, and positive integers, respectively. For a set $\mathcal{S}$, the cardinality of $\mathcal{S}$ is expressed as $|\mathcal{S}|$, and the power set of $\mathcal{S}$ is denoted as $\mathcal{P}(\mathcal{S}) = \{A : A \subseteq \mathcal{S}\}$. The empty set is indicated by $\emptyset$. In set operations, union, intersection, and complement are denoted by $\cup$, $\cap$, and $\setminus$, respectively.

For a function $f : X \to Y$, the image of a set $Q \subseteq X$ and the preimage of $P \subseteq X$ under f are denoted as $f(Q)$ and $f^{-1}(P)$, respectively. Denoting f^A indicates that the calculation follows the same mapping rules, but with different adjacency matrices A as part of the function argument.

The notation $\{1, 2, 3\}^n$ represents an integer vector of dimension n. Define vector $\boldsymbol{x} \subseteq \{1, 2, 3\}^n$, where x_i denotes the ith entry of $\boldsymbol{x}$. Logical operators OR, AND, and NOT are denoted by $\vee$, $\wedge$, and $\neg$, respectively.

The lexicographic cone is represented as $K_{lex} = \{0\} \cup \{\boldsymbol{x} \in \mathbb{R}^n : x_1 = \cdots = x_k = 0, x_{k+1} > 0\}$ for some $0 \le k < n$. The lexicographic ordering on $\mathbb{R}^n$ is denoted by $\boldsymbol{a} \le_{lex} \boldsymbol{b}$ if and only if $\boldsymbol{b} - \boldsymbol{a} \in K_{lex}$, where $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$.

2. Preliminaries and problem formulation

2.1. Graph theory and network robustness

An multiagent network with n agents can be represented by a directed graph (digraph for short) $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, 2, \ldots, n\}$ is a set of vertices and $\mathcal{E}$ is a set of edges. A directed edge is denoted as (i, j), indicating that agent i can send messages to agent j. The in-neighbors set of node i is represented by $\mathcal{N}_i = \{j \in \mathcal{V} : (j, i) \in \mathcal{E}\}$. The topology of a digraph is expressed as an adjacency matrix $A = [a_{ij}] \in R^{n\times n}$, where $a_{ij} = 1$ if agent i can receive information from agent j, and $a_{ij} = 0$ otherwise. A simple graph is a type of graph wherein self-edges and duplicated edges are not present. Formally, this means it adheres to the condition $(i, i) \notin \mathcal{E}, \forall i \in \mathcal{V}$ and $(i, j) \in \mathcal{E}$, indicating the existence of a singular directed edge from vertex i to vertex j. In this study, we focus on simple graphs as well as nonempty and nontrivial graphs, where $|\mathcal{V}| > 1$. Let $\mathcal{S}^{x^i}, i = \{1, 2, 3\}$ denote three disjoint subsets of $\mathcal{V}$ in $\mathcal{D}$, then we have $\mathcal{S}^{x^1} \cap \mathcal{S}^{x^2} = \emptyset$, $\mathcal{S}^{x^1} \cap \mathcal{S}^{x^3} = \emptyset$, $\mathcal{S}^{x^2} \cap \mathcal{S}^{x^3} = \emptyset$, and $\mathcal{S}^{x^1} \cup \mathcal{S}^{x^2} \cup \mathcal{S}^{x^3} = \mathcal{V}$.

Assumption 2.1. In this paper, the subsequent analysis focuses exclusively on non-empty, non-trivial, and simple digraphs.

Network robustness is a novel graph property that refers to the extent of redundancy in directed information flow between node subsets within a network [12]. This robustness specifically includes four metrics: r-reachable, r-robust, (r, s)-reachable, and (r, s)-robust, with their detailed definitions as follows.

Definition 2.1 (*r-reachable [12]*). In a digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with a non-negative integer $r \in \mathbb{Z}_+$, a non-empty subset $\mathcal{S} \subset \mathcal{V}$ is deemed r-reachable if there exists a vertex $i \in \mathcal{S}$ such that $|\mathcal{N}_i \setminus \mathcal{S}| \ge r$.

Definition 2.2 (*r-robust [12]*). For $r \in \mathbb{Z}_+$, a nonempty and nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with $n \geqslant 2$ nodes is said to be r-robust if at least one subset is r-reachable for every pair of nonempty and disjoint subsets of $\mathcal{V}$. Specifically, the empty graph ($n = 0$) and trivial digraph ($n = 1$) are considered 0-robust and 1-robust, respectively.

Definition 2.3 (*(r, s)-reachable [12]*). For a nonempty, nontrivial, and simple digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with $n \geqslant 2$ nodes, and $r, s \in \mathbb{Z}_+$ where $0 \le s \le n$, let $\mathcal{S}$ be a nonempty subset of $\mathcal{V}$, and define the set $\chi^r_{\mathcal{S}} = \{j \in \mathcal{S} : |\mathcal{N}_j \setminus \mathcal{S}| \ge r\}$. $\mathcal{S}$ is said to be (r, s)-reachable if there are at least s nodes in $\mathcal{S}$, and each of these nodes has at least r in-neighbors outside of $\mathcal{S}$. In other words, $\mathcal{S}$ is an (r, s)-reachable set if $|\chi^r_{\mathcal{S}}| \geqslant s$.

Definition 2.4 (*(r, s)-robust [12]*). For a nonempty, nontrivial, and simple digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with $n \geqslant 2$ nodes, and $r, s \in \mathbb{Z}_+$ where $0 \le s \le n$, we say that the digraph $\mathcal{D}$ is (r, s)-robust if at least one of the following conditions is satisfied for every pair of nonempty, disjoint subsets $\mathcal{S}_1, \mathcal{S}_2 \subseteq \mathcal{V}$:

$$\textbf{(A)} \left|\chi^r_{\mathcal{S}_1}\right| = \left|\mathcal{S}_1\right| \textbf{(B)} \left|\chi^r_{\mathcal{S}_2}\right| = \left|\mathcal{S}_2\right| \textbf{(C)} \left|\chi^r_{\mathcal{S}_1}\right| + \left|\chi^r_{\mathcal{S}_2}\right| \geqslant s. \tag{1}$$

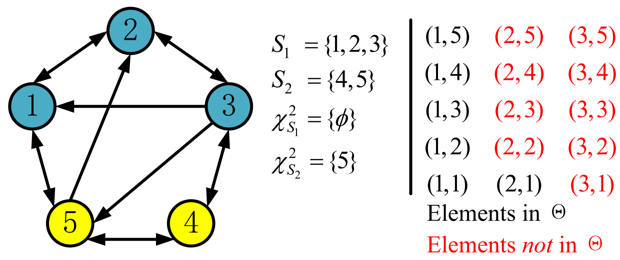

Fig. 1. The paragraph presents an example of the Θ for a given digraph $\mathcal{D}$ and its $(r^*,\ s^*)$-robustness. Since $|\mathcal{V}| = 5$, the possible $(r,\ s)$-robustness of $\mathcal{D}$ can be observed in the right-side illustration. The subsets $\mathcal{S}_1$ and $\mathcal{S}_2$ meet the conditions $\left|\chi^r_{S_1}\right| \neq |S_1|$, $\left|\chi^r_{S_2}\right| \neq |S_2|$, $\left|\chi^r_{S_1}\right| = 0$, and $\left|\chi^r_{S_2}\right| = 1$. Thus, the maximum possible element of Θ cannot reach (2,2). All elements of Θ are listed, and the $(r^*,\ s^*)$-robustness of $\mathcal{D}$ is determined as the maximum element (2,1).

According to Definitions 2.2 and 2.4, it should be noted that r-robustness is equivalent to $(r, 1)$-robustness.

2.2. Problem formulation

Most pre-existing resilient consensus algorithms [12,17,18,30] require the values of r and s in network robustness to be sufficiently large to ensure the system's convergence under the influence of a certain upper bound number of malicious agents. Definitions 2.1–2.4 clarify that evaluating the r- and (r, s)-robustness poses a combinatorial challenge, falling under the category of NP-hard problems. As a result, seeking an approximate solution for r- and (r, s)-robustness becomes a reasonable approach. It is noteworthy that digraph $\mathcal{D}$ may exhibit multiple satisfactory robustness properties. The following property clarifies that digraphs possess different robustness properties.

Property 2.1 ([12]). *A digraph $\mathcal{D}$ is considered $(r',\ s')$-robust if, for any r' where $0 \le r' \le r$ and any s' where $0 \le s' \le s$, $\mathcal{D}$ is $(r,\ s)$-robust.*

The robustness of digraph $\mathcal{D}$ is indeed characterized by backward compatibility. To clarify this feature, we provide the definition of the value space Θ.

$$\Theta = \left\{(r,s) \in \mathbb{Z}_+ : \forall(\mathcal{S}_1,\mathcal{S}_2) \in \tau, (\left|\chi^r_{\mathcal{S}_1}\right| = |\mathcal{S}_1|) \vee (\left|\chi^r_{\mathcal{S}_1}\right| = |\mathcal{S}_1|) \vee (\left|\chi^r_{\mathcal{S}_1}\right| + \left|\chi^r_{\mathcal{S}_2}\right| \ge s)\right\}. \tag{2}$$

The following definition represents the maximal (r, s)-robustness of a given digraph.

Definition 2.5. The maximum element of Θ under the lexicographical order on $\mathbb{R}^2$ is denoted to $(r^*,\ s^*)$.

According to Property 2.1 and Definition 2.5, it can be deduced that $(r^*,\ s^*)$ represents a digraph's maximum robustness. Indeed, to determine (r^*, s^*)-robustness of $\mathcal{D}$, one must first calculate the optimal value of r and then calculate the corresponding optimal value of s based on that r. For convenience of the following discussion, we denote $F_{max} = \max(\{F \in \mathbb{Z}_+ : (F+1, F+1) \in \Theta\})$.

This paper employs heuristic methods to approximate the robustness metrics of a digraph $\mathcal{D}$, including $r_{\max}(\mathcal{D})$, $(r^*,\ s^*)$-robustness, and $(F_{\max}+1,\ F_{\max}+1)$-robustness. The problems are formulated as follows.

Problem 1. Given a nonempty, simple digraph $\mathcal{D}$, determine the value of $r_{max}(\mathcal{D})$ approximatively.

Problem 2. Given a nonempty, simple digraph $\mathcal{D}$, determine the $(r^*,\ s^*)$-robustness of $\mathcal{D}$ approximatively.

Problem 3. Given a nonempty, simple digraph $\mathcal{D}$, determine the $(F_{max}+1,\ F_{max}+1)$-robustness of $\mathcal{D}$ approximatively.

3. Formalize determining *r*-robustness into a minimum problem of *n*-element discrete function

In this section, we propose a solution partition for addressing Problem 1, which is based on three vertex sets, and present the structure of the optimal solution partition. We then transform determining r into a minimum problem of an n-element discrete function using the structure. Given any nonempty, nontrivial, and simple digraph $\mathcal{D}$, we introduce a nonlinear n-element discrete function, the minimum value of which is equal to $r_{max}(\mathcal{D})$. Subsequently, we present an equivalent expression that clarifies the determination of $r_{max}(\mathcal{D})$ as an optimization problem based on two sets. For a nonempty, nontrivial, and simple digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ and a subset $\mathcal{S} \in \mathcal{V}$, we define a reachability function $\mathcal{R} : \mathcal{P}(\mathcal{V}) \to \mathbb{Z}_+$ as follows:

$$\mathcal{R}(\mathcal{S}) = \begin{cases} \max_{i\in\mathcal{S}} \left|\mathcal{N}_i \setminus \mathcal{S}\right|, & \text{if } \mathcal{S} \neq \{\emptyset\}, \\ 0, & \text{if } \mathcal{S} = \{\emptyset\}. \end{cases} \tag{3}$$

The result of (3) reflects the maximum r for which $\mathcal{S}$ is r-reachable. $r_{max}(\mathcal{D})$ can be expressed as an optimization formulation with (3) as follows.

Lemma 3.1 ([21]). *For a nonempty, nontrivial, and simple digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$. $r_{max}(\mathcal{D})$ can be defined as follows:*

$$\begin{aligned} &r_{max}(\mathcal{D}) = \min_{\mathcal{S}_1,\mathcal{S}_2\in\mathcal{P}(\mathcal{V})} \max(\mathcal{R}(\mathcal{S}_1), \mathcal{R}(\mathcal{S}_2)), \\ &\textit{subject to } |\mathcal{S}_1| > 0, |\mathcal{S}_2| > 0, |\mathcal{S}_1 \cap \mathcal{S}_2| = 0. \end{aligned} \tag{4}$$

Given a collection of nonempty, disjoint sets $\mathcal{S} \in \mathcal{P}(\mathcal{V})$, the following lemma presents their combination in relation to $\mathcal{R}(\mathcal{S})$.

Lemma 3.2. *Consider a nonempty, nontrivial, and arbitrary digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$. Let $\mathcal{S}_1, \mathcal{S}_2$ be subsets of $\mathcal{P}(\mathcal{V})$. The following holds:*

$$\begin{aligned} &\mathcal{R}(\mathcal{S}_1 \cup \mathcal{S}_2) \le \max\left\{\mathcal{R}(\mathcal{S}_1), \mathcal{R}(\mathcal{S}_2)\right\}, \\ &\quad \textit{subject to } |\mathcal{S}_1| > 0, |\mathcal{S}_2| > 0. \end{aligned} \tag{5}$$

Proof. Considering $\mathcal{S}_1$ and $\mathcal{S}_2 \in \mathcal{P}(\mathcal{V})$ satisfying the two constraints in (5), we assert that both sets are non-empty. The function $\mathcal{R}$ is defined by Eq. (3). Given $\mathcal{R}(\mathcal{S}_1) = r_0$ and $\mathcal{R}(\mathcal{S}_2) > r_0$, we can deduce the existence of an element $i \in \mathcal{S}_1$ such that $|\mathcal{N}_i \setminus \mathcal{S}_1| = r_0$. For any i, we have $|\mathcal{N}_i \setminus (\mathcal{S}_1 \cup \mathcal{S}_2)| \le r_0$. However, let $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2$, from which we obtain $|\mathcal{N}_i \setminus \mathcal{S}| = |\mathcal{N}_i \setminus (\mathcal{S}_1 \cup \mathcal{S}_2)| \le r_0$. Consequently, $\mathcal{R}(\mathcal{S}) = \mathcal{R}(\mathcal{S}_1 \cup \mathcal{S}_2) \le r_0 = \max\{\mathcal{R}(\mathcal{S}_1), \mathcal{R}(\mathcal{S}_2)\}$. When $\mathcal{R}(\mathcal{S}_2) = r_0 < \mathcal{R}(\mathcal{S}_1)$, we have the same derivation. Additionally, if $\mathcal{R}(\mathcal{S}_1) = \mathcal{R}(\mathcal{S}_2) = r_0$, for either $\forall i \in \mathcal{S}_1$ satisfying $|\mathcal{N}_i \setminus \mathcal{S}_1| = r_0$ or $\forall i \in \mathcal{S}_2$ satisfying $|\mathcal{N}_i \setminus \mathcal{S}_2| = r_0$, we can reach the same conclusion as $\mathcal{R}(\mathcal{S}) = \mathcal{R}(\mathcal{S}_1 \cup \mathcal{S}_2) \le r_0 = \max\{\mathcal{R}(\mathcal{S}_1), \mathcal{R}(\mathcal{S}_2)\}$. Therefore, we have $\mathcal{R}(\mathcal{S}_1 \cup \mathcal{S}_2) \le \max\{\mathcal{R}(\mathcal{S}_1), \mathcal{R}(\mathcal{S}_2)\}$. □

Employing Lemmas 3.1 and 3.2, we further investigate partitioning based on two sets. Then, we analyze partitioning scenarios from a problem-solving perspective. A specific partition can produce $r_{max}(\mathcal{D})$; we refer to these partitions as optimal partitions based on two sets (OPT2, for short). Based on Definition 2.2, we deduce the r-robustness of the two sets in OPT2 as detailed below.

Lemma 3.3. *For a nonempty and nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, there exists a partition in OPT2, denoted as*

$$\begin{aligned} &\mathcal{R}(\mathcal{S}_1) \le r, \mathcal{R}(\mathcal{S}_2) = r, \\ &\textit{subject to } |\mathcal{S}_1| > 0, |\mathcal{S}_2| > 0, |\mathcal{S}_1 \cap \mathcal{S}_2| = 0, \end{aligned} \tag{6}$$

if $\mathcal{D}$ satisfies r-robustness. Additionally, if $r_{max}(\mathcal{D}) = r$, it is impossible to find $\mathcal{S}_2$ within the formalization of OPT2, satisfying $\mathcal{R}(\mathcal{S}_2) = r - \varepsilon, \varepsilon \in \mathbb{Z}_+$.

The proof of Lemma 3.3 can be derived from Definition 2.2 and Lemma 3.1. It demonstrates that $\mathcal{S}_2$ in Lemma 3.3 represents the upper limit of potential r-robustness. The lowest achievable value of $\mathcal{R}(\mathcal{S}_2)$ determines $r_{\max}(\mathcal{D})$. For additional details, readers are directed to Ref. [12].

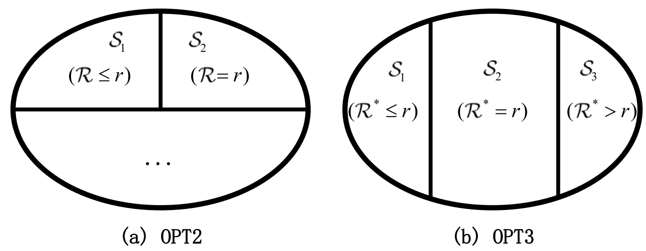

Fig. 2. Comparison between OPT2 and OPT3: The vertices of the digraph corresponding to r-robust are partitioned into sets, as shown in the figure. The sets $\mathcal{R}$ and $\mathcal{R}^*$ are defined by Eqs. (3) and (7), respectively. In OPT2, there exist only two non-empty and disjoint sets, whereas in OPT3, the vertices are divided into three disjoint sets, completing the partition.

The conclusion is based on two sets, suggesting that the union of sets $\mathcal{S}_1$ and $\mathcal{S}_2$ may not cover all vertices in $\mathcal{D}$. In other words, each vertex in this context possesses the dual state of either being selected or unselected and being incorporated into sets $\mathcal{S}_1$ or $\mathcal{S}_2$. As a result, an accurate depiction of OPT2 entails the use of $2n$ binary variables.

To describe the partition using n variables, we next discuss partitioning based on three sets. Similarly, we refer to these partitions, which yield $r_{max}(\mathcal{D})$, as optimal partitions based on three sets (briefly OPT3). For formalization purposes, the new function $\mathcal{R}^* : \mathcal{P}(\mathcal{S}) \to \mathbb{Z}_+$ is defined as follows.

$$\mathcal{R}^*(\mathcal{S}) = \begin{cases} \max\limits_{i \in \mathcal{S}} \left|\mathcal{N}_i \setminus \mathcal{S}\right|, & if\ \mathcal{S} \neq \{\emptyset\}, \\ |\mathcal{V}|, & if\ \mathcal{S} = \{\emptyset\}. \end{cases} \tag{7}$$

The following lemma provides a formal partitioning of the problem solution into three sets.

Lemma 3.4. *For a nonempty and nontrivial digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$, there exists a partition in OPT3, denoted as*

$$\begin{aligned} &\mathcal{R}^*(\mathcal{S}_1) \le r, \quad \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) > r, \\ &subject\ to \quad \left|\mathcal{S}_i \cap \mathcal{S}_j\right| = 0, i \neq j, \\ &\qquad\qquad\quad \left|\mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3\right| = |\mathcal{V}|, \end{aligned} \tag{8}$$

if $\mathcal{D}$ satisfies the property of r-robustness. Additionally, it is impossible to find $\mathcal{S}_2$ within the formalization of OPT3 that satisfies $\mathcal{R}^*(\mathcal{S}_2) = r - \varepsilon, \varepsilon \in \mathbb{Z}_+$ when $r_{max}(\mathcal{D}) = r$.

Proof. The proof consists of two parts. In the first part, we analyze the sufficiency of the above conditions, and in the second part we analyze their necessity.

Regarding the first part, suppose we have a digraph $\mathcal{D}$. Let $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3 \in \mathcal{P}(\mathcal{V})$, where $\mathcal{V}$ is the set of vertices, with the property that $\left|\mathcal{S}_i \cap \mathcal{S}_j\right| = 0$ for $i \neq j$, and $\left|\mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3\right| = |\mathcal{V}|$. The function $\mathcal{R}^*$ is defined by Eq. (7). If the OPT3 of $\mathcal{D}$ is denoted as $\mathcal{R}^*(\mathcal{S}_1) \le r$, $\mathcal{R}^*(\mathcal{S}_2) = r$, and $\mathcal{R}^*(\mathcal{S}_3) > r$, it implies the impossibility of finding any $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3 \in \mathcal{P}(\mathcal{V})$ that satisfy the constraints of Lemma 3.4 and $\mathcal{R}^*(\mathcal{S}_2) < r$. The further analysis is then extended based on whether the subsets $\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3$ are empty. Obviously, since we are considering non-empty digraphs in this paper, there is no case where all three sets are empty sets. If two subsets are empty, the remaining subset must be the universal set. This implies $\mathcal{R}^*(\mathcal{S}_1) = 0, \mathcal{R}^*(\mathcal{S}_2) = \mathcal{R}^*(\mathcal{S}_3) = |\mathcal{V}|$. Clearly, this situation does not satisfy OPT3. If only one subset is empty, it implies that $\mathcal{R}^*(\mathcal{S}_1) = \mathcal{R}(\mathcal{S}_1), \mathcal{R}^*(\mathcal{S}_2) = \mathcal{R}(\mathcal{S}_2), \mathcal{R}^*(\mathcal{S}_3) = |\mathcal{V}|$. Based on Lemma 3.2, we can deduce that $\mathcal{R}^*(\mathcal{S}_1) \le r, \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) = |\mathcal{V}| > r$. If none of the three subsets is empty, we have $\mathcal{R}^*(\mathcal{S}_1) = \mathcal{R}(\mathcal{S}_1), \mathcal{R}^*(\mathcal{S}_2) = \mathcal{R}(\mathcal{S}_2), \mathcal{R}^*(\mathcal{S}_3) = \mathcal{R}(\mathcal{S}_3)$. This indicates that it is equivalent to Lemma 3.2, that is, $\mathcal{R}^*(\mathcal{S}_1) \le r, \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) > r$.

Regarding the second part, suppose there is a digraph $\mathcal{D}$ such that $r_{max}(\mathcal{D}) = r$. As $\mathcal{D}$ satisfies $r_{max}(\mathcal{D}) = r$, we need to identify a nonempty subset $\mathcal{S} \in \mathcal{P}(\mathcal{V})$ such that $\mathcal{R}^*(\mathcal{S}) = \mathcal{R}(\mathcal{S}) = r$. We fix subset $\mathcal{S}$ and explore the partition of the remaining sets. It is evident that in OPT3, only one of the three subsets can be empty at most. By applying Lemma 3.3, we can delineate a partition of three sets, ensuring that $\mathcal{R}(\mathcal{S}_1) \le r$, $\mathcal{R}(\mathcal{S}_2) = r$, and $\mathcal{R}(\mathcal{S}_3) \in (0, |\mathcal{V}|)$. Consequently, we obtain that $\mathcal{R}^*(\mathcal{S}_1) \le r, \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) \in (0, |\mathcal{V}|)$. We now focus on $\mathcal{R}^*(\mathcal{S}_3)$. When $\mathcal{R}^*(\mathcal{S}_3) < r$ (assuming $\mathcal{R}^*(\mathcal{S}_1) = \mathcal{R}^*(\mathcal{S}_2) = r$), we merge $\mathcal{S}_1$ with $\mathcal{S}_3$ as $\mathcal{S}_1$. Then, $\mathcal{S}_3$ is the empty subset and we can deduce that $\mathcal{R}^*(\mathcal{S}_1) = \mathcal{R}(\mathcal{S}_1) \le r$ from Lemma 3.2. This implies that $\mathcal{R}^*(\mathcal{S}_1) \le r, \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) = |\mathcal{V}| > r$. When $\mathcal{R}^*(\mathcal{S}_3) \ge r$, we have $\mathcal{R}^*(\mathcal{S}_1) \le r, \mathcal{R}^*(\mathcal{S}_2) = r, \mathcal{R}^*(\mathcal{S}_3) \ge r$. □

An example of OPT2 and OPT3 discussed above is illustrated in Fig. 2. According to Lemma 3.4, finding the minimum of the following n-element discrete function is equivalent to accurately solving Problem 1.

Lemma 3.4 asserts that determining the minimum value of the discrete function comprising n elements equates to accurately solving Problem 1.

Theorem 3.1. *For a nonempty, nontrivial, and arbitrary digraph $\mathcal{D}$, the maximum r-robustness, denoted as $r_{max}(\mathcal{D})$, is determined by finding the minimum value of the discrete function $f^A(\boldsymbol{x})$ which is define as*

$$f^A(\boldsymbol{x}) = mid(\mathcal{R}^*(\mathcal{S}^{x^1}), \mathcal{R}^*(\mathcal{S}^{x^2}), \mathcal{R}^*(\mathcal{S}^{x^3})), \tag{9}$$

where A represents the adjacency matrix of $\mathcal{D}$. In other words, we have

$$r_{max}(\mathcal{D}) = min\left\{f^A(\boldsymbol{x})\right\}. \tag{10}$$

Proof. Consider a nonempty, nontrivial, and arbitrary digraph $\mathcal{D}$, and examine its OPT3 property. Based on Lemma 3.4, we conclude that $\mathcal{R}^*(\mathcal{S}_2)$ in OPT3 is equal to $r_{max}(\mathcal{D})$ of $\mathcal{D}$. However, the search space of $\mathcal{D}$, derived from three sets, is 3^n. In our partition from the general form to OPT3, $\mathcal{R}^*(\mathcal{S}_2)$ represents the global minimum. Then, from (8), we deduce

$$\mathcal{R}^*(\mathcal{S}_2) \le \mathcal{R}^*(\mathcal{S}_{x2}), \tag{11}$$

where $\mathcal{S}_{x2}$ is the second subset of the general form partition, ordered by $\mathcal{R}^*(\mathcal{S})$. Subsequently, the definitions of $\mathcal{S}^x$ and $\mathcal{R}^*(\mathcal{S}^x)$ imply that, in (9), $mid(\mathcal{R}^*(\mathcal{S}^{x^1}), \mathcal{R}^*(\mathcal{S}^{x^2}), \mathcal{R}^*(\mathcal{S}^{x^3}))$ equals $\mathcal{R}^*(\mathcal{S}_{x2})$. Therefore, when we compute the minimum of (9), we obtain $r_{max}(\mathcal{D})$ for $\mathcal{D}$. □

4. Formalizing determining (*r*, *s*)-robustness into a minimum problem of *n*-element discrete function

In this section, we discuss the structure of determining (r, s)-robustness with partition-based solutions and transform determining s into a minimum problem of n-element discrete function for solving Problem 2 based on three sets. For further discussion, we use the following notation:

Definition 4.1. Given a digraph $\mathcal{D}$ and $r \in \mathbb{Z}_+$, the maximum integer s which satisfies (r, s)-robustness is denoted as $s_{max}(r) \in \mathbb{Z}_+$. Specifically, we say $s_{max}(r) = 0$ if the digraph $\mathcal{D}$ cannot reach r-robustness.

With this notation, what we aim for next, as mentioned in Problem 2, satisfies $r = r_{max}(\mathcal{D})$ and $s = s_{max}(r_{max}(\mathcal{D}))$. It is evident that (r, s) is the maximum element of Θ described in (2) by lexicographic ordering. We have previously shown that solving r and solving s for which a given digraph satisfies (r, s)-robustness have a topological order. A contribution to determining r-robustness has already been presented. In this section, we discuss how to describe solving $s_{max}(r)$ for any given $r \in \mathbb{Z}_+$ based on three-set partition. Subsequently, it can be used to obtain $s_{max}(r_{max}(\mathcal{D}))$ after $r_{max}(\mathcal{D})$ is determined. More explicitly, $s_{max}(r)$ can be given using the following notation:

Definition 4.2. For Θ as the set of (r, s) values for which a given digraph $\mathcal{D}$ satisfies (r, s)-robustness, $r \in \mathbb{Z}_+$ and $\chi_{\mathcal{S}}^r$ is denoted as in Definition 2.4, define a set Θ_r as follows:

$$\Theta_r = \{s \in \mathbb{Z}_+ : \forall(\mathcal{S}_1, \mathcal{S}_2) \in \tau, (\left|\chi_{\mathcal{S}_1}^r\right| = |\mathcal{S}_1|) \vee (\left|\chi_{\mathcal{S}_1}^r\right| = |\mathcal{S}_1|) \vee (\left|\chi_{\mathcal{S}_1}^r\right| + \left|\chi_{\mathcal{S}_2}^r\right| \geq s)\}. \tag{12}$$

Analogously, Θ_r is the set of $s \in \mathbb{Z}_+$ for which a given digraph $\mathcal{D}$ satisfies (r, s)-robustness, similar to Θ. The maximum element of Θ_r is equal to $s_{max}(r)$. The three constraints in the RHS of (12) correspond to conditions (**A**), (**B**), (**C**) of Definition 2.4 sequentially. Note that these three conditions are connected by $\vee$, indicating logical OR. In other words, we have to accept $\mathcal{S}$ satisfied $\left|\chi_{\mathcal{S}}^r\right| = |\mathcal{S}|$ as partition results, even though these results do not yield a valid s. This implies that the selected set pairs mask s that we want to determine.

Before resolving this problem, we define a reachability function $\chi^* : (\mathcal{P}(\mathcal{V}), r) \to \mathbb{Z}$ as follows:

$$|\chi^*(\mathcal{S}, r)| = \begin{cases} \left|\chi_{\mathcal{S}}^r\right|, & if\ \mathcal{S} \neq \{\emptyset\}, \\ |\mathcal{V}|, & if\ \mathcal{S} = \{\emptyset\}. \end{cases} \tag{13}$$

Using this notation, we demonstrate the partition of the form of the solution based on three sets if it exists in the following lemma:

Lemma 4.1. *Given a nonempty, nontrivial, and arbitrary digraph $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ and an integer $s \in \mathbb{Z}$, there is a division denoted such that*

$$\begin{aligned} &|\chi^*(\mathcal{S}_1, r)| = min\left\{|\chi(\mathcal{S}_1, r)|, |\chi(\mathcal{S}_2, r)|, |\chi(\mathcal{S}_3, r)|\right\}, \\ &|\chi^*(\mathcal{S}_2, r)| = mid\left\{|\chi(\mathcal{S}_1, r)|, |\chi(\mathcal{S}_2, r)|, |\chi(\mathcal{S}_3, r)|\right\}, \\ &|\chi^*(\mathcal{S}_3, r)| = max\left\{|\chi(\mathcal{S}_1, r)|, |\chi(\mathcal{S}_2, r)|, |\chi(\mathcal{S}_3, r)|\right\}, \\ &subject\ to \left|\mathcal{S}_i \cap \mathcal{S}_j\right| = 0, i \neq j\ |\mathcal{S}_1 \cup \mathcal{S}_2 \cup \mathcal{S}_3| = |\mathcal{V}|. \end{aligned} \tag{14}$$

By Definition 2.4, we transform solving for s into a programming problem subject to constraints, as follows:

Lemma 4.2. *Given a nonempty, nontrivial, arbitrary digraph $\mathcal{D}$ and $r \in \mathbb{Z}_+$, the maximum integer s, which satisfies (r, s)-robustness, is determined by solving the minimum of the following n-element discrete function:*

$$\begin{aligned} &h(\boldsymbol{x}, r) &&= |\chi^*(\mathcal{S}_1, r)| + |\chi^*(\mathcal{S}_2, r)|, \\ &subject\ to && |\chi^*(\mathcal{S}_1, r)| < |\mathcal{S}_1|, |\chi^*(\mathcal{S}_2, r)| < |\mathcal{S}_2|. \end{aligned} \tag{15}$$

If (15) does not exist, the minimum of $h(\boldsymbol{x}, r)$ is equal to $|\mathcal{V}|$.

Proof. Our proof depends on (12). Recall that the maximum element of Θ_r is equal to $s_{max}(r)$. So we just have to prove that the minimum value of (15) is equal to the maximum element of Θ_r. The fact that (15) does not exist means each partition $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$ of $\mathcal{D}$ in Lemma 4.1 satisfies $|\chi^*(\mathcal{S}_1, r)| = |\mathcal{S}_1|$ *and* $|\chi^*(\mathcal{S}_2, r)| = |\mathcal{S}_2|$. So the maximum in (12) is equal to $|\mathcal{V}|$. The result is obvious. Next we need to prove it when (15) does exist.

The proof begins by discussing whether $\{\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_3\}$ are empty. Three subsets cannot be empty sets at the same time obviously. When having two subsets empty, the other subset is $\mathcal{V}$. Denote that $\{\mathcal{S}_1 = \mathcal{V}, \mathcal{S}_2 = \emptyset, \mathcal{S}_3 = \emptyset\}$. We have

$$|\chi^*(\mathcal{S}_1, r)| = 0, |\chi^*(\mathcal{S}_2, r)| = |\mathcal{V}|, |\chi^*(\mathcal{S}_3, r)| = |\mathcal{V}|. \tag{16}$$

This partition is going to be equal to $|\mathcal{V}|$, which obviously not what we are looking for; When having only one subset empty, we denote that $\{\mathcal{S}_1 \neq \emptyset, \mathcal{S}_2 \neq \emptyset, \mathcal{S}_3 = \emptyset\}$. Causing the nonempty subsets found out subject to $|\chi^*(\mathcal{S}_1, r)| < |\mathcal{S}_1|, |\chi^*(\mathcal{S}_2, r)| < |\mathcal{S}_2|$, the subsets $\mathcal{S}_1, \mathcal{S}_2$ satisfy the constraint 3 in the RHS of (12). In other words, we can obtain the value of s from the partition if we find the value that minimizes $h(\boldsymbol{x}, r)$ under this condition. While (12) is such that all pairs satisfy the constraint, so that is the maximum number of elements in set Θ_r; When having no one subset empty, we denote that $\{\mathcal{S}_1 \neq \emptyset, \mathcal{S}_2 \neq \emptyset, \mathcal{S}_3 \neq \emptyset\}$. Using Lemma 4.1 to round out this partition, we have $|\chi^*(\mathcal{S}_1, r)| \leq |\chi^*(\mathcal{S}_2, r)| \leq |\chi^*(\mathcal{S}_3, r)|$. In this case, we treat $\mathcal{S}_1, \mathcal{S}_2$ as $\mathcal{S}_1, \mathcal{S}_2$ in (12) subject to $|\chi^*(\mathcal{S}_1, r)| < |\mathcal{S}_1|, |\chi^*(\mathcal{S}_2, r)| < |\mathcal{S}_2|$. So we can come to the same conclusion that when we obtain the minimum of $h(\boldsymbol{x}, r)$, it is the maximum number of elements in set Θ_r. □

Based on (15), we propose the following method for computing $s_{max}(r_{max}(\mathcal{D}))$:

Theorem 4.1. *Consider a nonempty, nontrivial, and arbitrary digraph $\mathcal{D}$ with $r \in \mathbb{Z}_+$. Solving for the minimum of (15) is equivalent to finding the minimum of the function below when $M_k \geq \frac{|\mathcal{V}|}{2}$:*

$$\begin{aligned} g^k(\boldsymbol{x}, r) = h(\boldsymbol{x}, r) + M_k \textstyle\sum_{i=1}^2 [&|\chi^*(\mathcal{S}_i, r)| - |\mathcal{S}_i| + 1 \\ &+ ||\chi^*(\mathcal{S}_i, r)| - |\mathcal{S}_i| + 1|], \end{aligned} \tag{17}$$

which means we obtain

$$s_{max}(r) = \min\left\{g^k(\boldsymbol{x}, r)\right\}. \tag{18}$$

Proof. By Lemma 4.2, we know that the minimum value for $h(\boldsymbol{x}, r)$ is $s_{max}(r)$ of $\mathcal{D}$ as long as the constraints in (15) is satisfied. So what we need to prove is that the minimum of (17) is equal to the minimum of (15) within constraints. (15) requires that $|\chi^*(\mathcal{S}_1, r)| < |\mathcal{S}_1|, |\chi^*(\mathcal{S}_2, r)| < |\mathcal{S}_2|$. This is because if at least one of the two constraints is not valid, it can be known from Definition 4.2 that such partition cannot obtain the value of s. Then if we expect the minimum value of (17) to be equal to s, it is very necessary to guarantee that the minimum will not be taken when the constraints equivalence is not satisfied. To describe it explicitly, we next denote that $\mathcal{A}(\mathcal{S}) = |\chi^*(\mathcal{S}, r)| - |\mathcal{S}| + 1, \mathcal{S} \in \mathcal{P}(\mathcal{V})$. Recall that $\chi(\mathcal{S}, r) \in \mathcal{S}$ and $\chi(\mathcal{S}, r) = \chi^*(\mathcal{S}, r)$ when $\mathcal{S} \neq \emptyset$. So we have $|\chi^*(\mathcal{S}, r)| \leq |\mathcal{S}|$. Furthermore, it can derive that $|\chi^*(\mathcal{S}, r)| - |\mathcal{S}| \leq 1$, and the equal sign can be taken if and only if $\chi^*(\mathcal{S}, r) = \mathcal{S}$. $|\chi^*(\mathcal{S}, r)| - |\mathcal{S}| = 1$ means that the constraint of (15) is violated. So we can obtain that $g^k(\boldsymbol{x}, r) \geq M_k(\mathcal{A} + |\mathcal{A}|)$. When $\mathcal{A} = 1$ and $M_k \geq \frac{|\mathcal{V}|}{2}$, we have $g^k(\boldsymbol{x}, r) \geq |\mathcal{V}|$. So $g^k(\boldsymbol{x}, r)$ is definitely not going to be the minimum in this case. In other words, the minimum value of $g^k(\boldsymbol{x}, r)$ is obtained only if it does not violate the constraint. That means that the minimum of (17) is equal to the minimum of (15) within constraints. □

5. Approximate solution of *n* -element discrete function minimum using genetic algorithm

In this section, we discuss the *n*-element discrete function introduced in Sections 3 and 4. We then design a Genetic Algorithm (GA) [31] to approximately solve the complex implicit functions.

5.1. Function analysis

Recall the implicit function presented in Sections 3 and 4, which represents the minimum *r*-robustness and (r, s)-robustness for a given integer $r \in \mathbb{Z}_+$ and a digraph $\mathcal{D}$. However, acquiring the minimum poses an intractable problem. Indeed, previous work has not altered the structure of Problems 1 and 2, indicating that finding the function minimum remains a combinatorial, NP-hard problem. Thus, we cannot determine it in polynomial time.

This observation suggests that the function's structure need not be considered extensively when employing heuristic algorithms to solve the function minimum or maximum. Moreover, the reduced number of variables ($2n \to n$) enhances the efficiency for solving clusters of similarly structured functions. This characteristic allows us to select a suitable heuristic algorithm to address the problem.

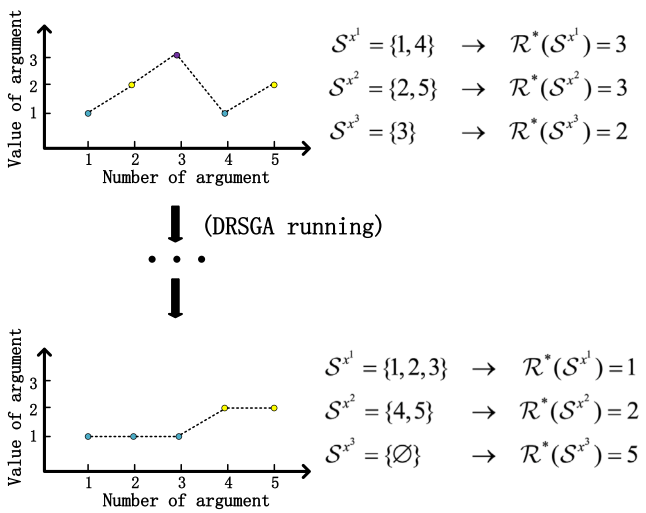

Fig. 3. An example of the DRSGA running process to determine r-robustness of $\mathcal{D}$ in Fig. 1. The abscissa represents the number of each point in $\mathcal{D}$, and it also illustrates a gene sequence from an initial random population $I_i = (1,2,3,1,2)$ or the argument value $\boldsymbol{x} = (1,2,3,1,2)$ of the function. The graph below shows the argument value $\boldsymbol{x} = (1,2,3,1,2)$ that might appear at the end of the algorithm run. The value of function (9) is changed from 3 to 2.

5.2. Outline algorithm

Although all heuristic algorithms can be employed to solve minimizing function problems [32], each method has unique advantages. GA, an optimization method inspired by evolution and natural selection principles, is a heuristic search algorithm used to approximate solutions to optimization problems. We aim to obtain considerable parallel computing power and preserve as many algorithm solutions as possible in the results from GA, even if it sacrifices some tolerable computing time.

The proposed algorithm, DRSGA, is formulated with GA support, which includes *Selection*, *Crossover*, and *Mutation* operators. The algorithm's effectiveness relies on the synergy among these operators. Additionally, the length of the gene code plays a crucial role in determining the efficiency of the GA. In DRSGA, the parameter I signifies the gene code length utilized in the algorithm. Each gene symbolizes an individual, and assemblages of individuals constitute the group P. To enhance the algorithm's clarity, we represent the group as a set $P = \{I_1, I_2, \ldots, I_n\}$. The primary objective of DRSGA is to attain the individual having the highest fitness value $Fit(I_i)$ following the mapping rule defined by Eq. (19).

5.3. The DRSGA algorithm

Under the synergistic action of the three operators, our algorithm approximates the minimum values of functions (9) and (17) through two independent iterations. The input of the algorithm is the entry adjacency matrix A of the digraph $\mathcal{D}$. The return value of the algorithm is the approximate (r^*, s^*)-robustness of the digraph $\mathcal{D}$, and a possible partition result that determines the values of the two. Each operator is described as follows:

5.3.1. Initialization

Typically, the algorithm begins with a random population. For instance, we can initialize the population using the algorithm of random generation of chromosomes. This is because DRSGA aims to ensure chromosome diversity in the population during the early and middle stages of operation. The powerful global parallel search capability of the algorithm benefits from this diversity. When the algorithm lacks sufficient differential genes in the early stage, it often converges to the local optimal solution prematurely, and it is challenging to escape this state through the *Mutation* process.

However, functions (9) and (17) have a highly complex solution space structure. As a result, even with a fully random population initialization, the subsequent iteration process is prone to converge to the local optimal solution. Thus, as a rule of thumb, we can try to retain some of the "better genes" during population initialization.

From the analysis of partition division results for solving Problem 1 based on three sets, it can be observed that points with large entry degrees are often divided into $\mathcal{S}_3$ described by (8).

Therefore, when initializing the population for solving Problem 1, we chose to initialize 80% of the population chromosomes randomly, and assigned the variables corresponding to the nodes with high degree of inclusion to 3 for the remaining 20%, while the rest were assigned random values between 1 and 2.

5.3.2. Selection

In natural biological populations, a phenomenon occurs where individuals with strong adaptability have a greater probability of obtaining resources necessary for survival, enabling them to pass on their superior genes. The Genetic Algorithm (GA) is inspired by this process of natural selection. Drawing from this concept, our algorithm employs the roulette method to select individuals in cases where there is a positive correlation between the fitness value of population individuals and probability, in a manner similar to the GA algorithm.

First, we introduce the roulette strategy. The roulette strategy comprises a roulette wheel with a segmented fan area of varying sizes based on the number of individuals and a fixed pointer directed toward the center of the wheel. Each selection involves rotating the roulette wheel, causing it to stop randomly, at which point the pointer will point to a sector representing an individual. This indicates that the individual has been selected for further processing.

With a positive correlation between individual fitness values and sector area sizes, this method ensures that individuals with higher fitness values are more likely to be selected for the subsequent Crossover process.

Next, we describe how we combine individual fitness with the probability of being selected. The probability of an individual being selected should be reflected by their fitness value, which is calculated using the fitness value function. As we aim to solve for the minimum value of both functions, we can design a single fitness function. For clarity, we denote the function to be solved as $\sigma(\boldsymbol{x})$, and a fitness function $Fit : \boldsymbol{x} \to \mathbb{R}$ as follows:

$$Fit(\boldsymbol{x}) = \frac{|\mathcal{V}|}{\sigma(\boldsymbol{x}) + 1}. \tag{19}$$

Clearly, smaller function values will result in higher adaptation values after mapping through (19). We use probability accumulation to relate fitness value to the probability of being selected. First, we calculate the proportion of each individual's fitness value as follows:

$$p_i = \frac{Fit(\boldsymbol{x}_i)}{\sum_{j=1}^{n} Fit(\boldsymbol{x}_j) + 1}. \tag{20}$$

We then obtain the probability accumulation order as follows:

$$p_i' = \sum_{j=1}^{i} p_j. \tag{21}$$

Furthermore, it can be easily derived that $p_i' \in [0,1]$. Finally, we use the probability interval to simulate the roulette wheel and generate a random decimal between 0 and 1 to simulate the process of rotational selection between the pointer and the roulette wheel.

5.3.3. Crossover (recombination)

The primary goal of the algorithm is to generate individuals with high fitness values and eliminate those with low fitness values, ensuring the improvement of individual fitness while maintaining a constant population number. After selecting high-fitness individuals, it is crucial to produce offspring possessing enhanced genes through these selected individuals. In nature, species produce new offspring by exchanging genes between parents, increasing the likelihood of preserving superior genes. Similarly, our DRSGA emulates the generation of new individuals through recombination.

The *Crossover* operator serves to generate new individuals. Generally, there are various types of crossover methods; in this case, we employ *multi-point crossover* with a probability of P_c that each point will be exchanged. Additionally, P_c is referred to as the crossover rate.

5.3.4. Mutation

The *Mutation* operator is the final evolutionary component of DRSGA. This operator exists to address the algorithm's tendency to prematurely converge and fall into local optimal solutions during later stages. Introducing the mutation operator enables the algorithm to escape local optima.

All offspring are randomly assigned to $\left\lfloor \frac{|\mathcal{V}|}{5} \right\rfloor + 1$ gene sites with a mutation rate of P_m probability. Typically, P_m assumes very low values to prevent high P_m from reducing DRSGA to a rudimentary random search algorithm. The existence of mutation operators can increase a population's genetic diversity to a certain extent, thereby improving the global parallel search capability of DRSGA.

Algorithm 1 DRSGA

Initialize $P(0)$ ▷ $P(t)$ is the state of the population at time t.
for $i = 1$ to $|P|$ **do**
  Evaluate fitness of $P(t)$ based given function
$t = 0$
while $t \le T$ **do**
  for $j = 1$ to $|P|$ **do**
    Select operation in $P(t)$ to select two individuals
    Crossover operation to the individuals
    Add the two newly created individuals to P
  for $i = 1$ to $|P|$ **do**
    Mutation operation to $P(t)$
  for $i = 1$ to $|P|$ **do**
    Evaluate fitness of $P(t)$
  Eliminate $|P|$ individuals with low fitness value
  $t = t + 1$
return The function value corresponding to the individual with the maximum fitness

As a simple illustration, Fig. 3 demonstrates the process of approximately calculating the value of r. The algorithm's effectiveness hinges on the proper configuration of parameters, such as population size $|P|$, crossover rate P_c, mutation rate P_m, and iteration count T. Moreover, the genetic representation of each individual in the population after executing DRSGA can function as the optimal partition solution that the algorithm is capable of discovering. The subsequent sections provide an in-depth examination of the algorithm.

5.4. Algorithm analysis

While DRSGA cannot guarantee a theoretically provable level of precision, it consistently produces results that are as optimal as possible and often surpasses this threshold. Further conclusions are outlined below:

Theorem 5.1. *For a nonempty, nontrivial, and simple digraph $\mathcal{D}$, DRSGA's approximate determination of r-robustness and (r, s)-robustness for $\mathcal{D}$ possesses the following properties:*

1. *The results of DRSGA are not inferior to the exact solution of $r_{max}(\mathcal{D})$ and $s_{max}(r)$;*
2. *it operates within a $O(n^3)$ time complexity.*

Proof. For (1), we denote the set $(r_{max}(\mathcal{D}), s_{max}(r))$-reachable for all pairs of $\mathcal{D}$ as Ω, where $\Omega \subset \mathbb{Z}_+ \times \mathbb{Z}_+$. What we need to prove is that the maximum element of Θ denote Θ_{max} is equal to the minimum element of Ω under the lexicographical order on $\mathbb{R}^2$. By Definition 2.5, we know that Θ_{max} is $(r_{max}(\mathcal{D}), s_{max}(r))$ of $\mathcal{D}$. From Theorem 3.1 and Lemma 4.1 we can derive the lowest value that can be obtained by partition is equal to the minimum element of Ω. So when functions (9) and (17) are not minimized, it returns an approximate upper bound of $(r_{max}(\mathcal{D}), s_{max}(r))$, equal if and only if they are minimized.

For (2), Recall our DRSGA. It is actually an enhancement of the genetic algorithm. We examine each operator **in one iteration** from which DRSGA operates. In *Initialization*, it needs to initialize every gene for every individual in the population. So the time it takes for this phase to run is $T_1(nP)$ where P is population size; In *Selection*, it needs to calculate the fitness of each individual and select the two individuals with high fitness P times. The running time required to complete the tasks is $T_2(2n^3P)$; In *Crossover*, it involves crossing two strings of genes, which will finish in $T_3(2nP)$; In *Mutation*, it makes probabilistic variations on genes, which completes in $T_4(nP)$. So the total time it takes to run DRSGA is

$$T_1 + T_2 + T_3 + T_4 = T(2P(n^3 + 2n)). \tag{22}$$

We treat the parameter P as a constant, so there is a result that $O(n^3)$. □

Remark 5.1. In fact, for any values taken within their respective domains, (9) and (17) each correspond to a practical set partition. They can both yield an upper bound for either $r_{max}(\mathcal{D})$ or $s_{max}(r)$. The DRSGA, through the collaborative action of several operators, conducts parallel searches, aiming to identify the optimal partition as much as possible. It is noteworthy that DRSGA does not guarantee avoidance of falling into local optima, which may result in the returned solution being greater than the exact value. However, the *Mutation* operator, through mutation, enables the algorithm to some extent to escape local optima. In addition, the *Initialization* operator is specifically tailored for the assessment of both $r_{max}(\mathcal{D})$ and $s_{max}(r)$, enabling a customized configuration. This increases the likelihood of individuals with favorable performance appearing in *Initialization*.

5.5. Determining F_{max}

Theorem 4.1 states that the minimum value of function (17) equals s for a given r, thereby facilitating the resolution of Problem 3. To ascertain the $(F_{max}+1, F_{max}+1)$-robustness of a nonempty, nontrivial, and arbitrary digraph, note that $F_{max} = max(\{F \in \mathbb{Z}_+ : (F+1, F+1) \in \Theta\})$. We initiate by setting $(F_{max}+1)$ to the minimum value of function (9). Explicitly, the minimum of function (9) denotes the largest value of r for the digraph. Recall that for a given digraph $\mathcal{D}$, it satisfies (r, s) for all $(r', s') \in \Theta$, such that $r = s$. We then compare the minimum value of function (17) to $(F_{max}+1)$. If the former is no less than the latter, it implies that $F_{max} = r-1$. However, if $r > s$, the value of r is decremented and s must be recalculated. The algorithm terminates when the largest value r renders $\mathcal{D}$ (r, r)-robust, returning $F_{max} = (r-1)$.

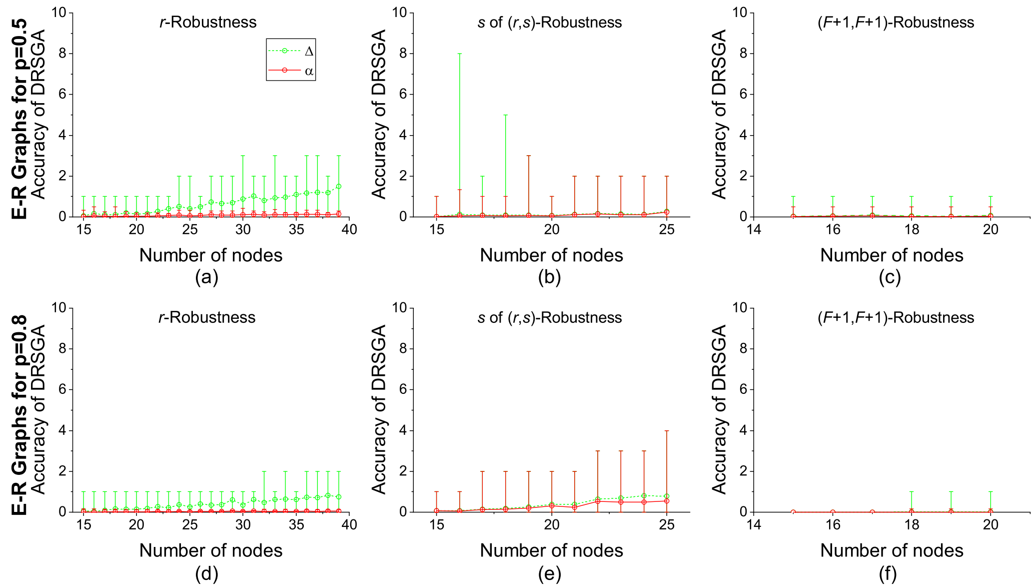

Fig. 4. Illustration of DRSGA's accuracy. The approximate result generated by the algorithm is guaranteed to be no less than the exact value obtained through MILP, ensuring the accuracy of the approach. The accuracy of the algorithm is quantified using $\Delta = R_1 - R_2$ and $\alpha = \frac{R_1 - R_2}{R_2}$ for cases (a)(b)(d)(e), whereas for cases (c)(f), it is measured using $\Delta = R_1 - R_2$ and $\alpha = \frac{R_1 - R_2}{R_2 + 1}$. In the figure, lines and circles represent the average difference between the obtained solution and the actual value, calculated over 100 digraphs for each number of n. The lower and upper lines represent the minimum and maximum value of the difference, respectively. Importantly, all of these differences have non-negative values.

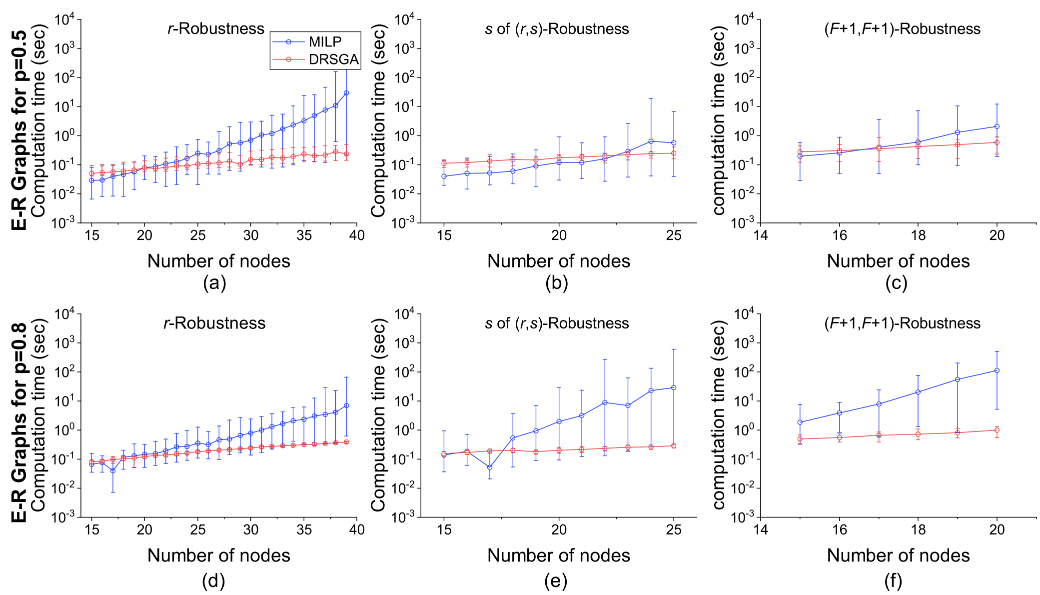

Fig. 5. Comparison of DRSGA and MILP formulations. The lines and circles in the figure represent the average computation time over 100 digraphs for each value of n, with the lower and upper lines indicating the minimum and maximum computation times, respectively. Approximate determination of r-robustness, (r, s)-robustness for a given r, and $(F+1, F+1)$-robustness are successively displayed. It is important to note that DRSGA ceases execution after a fixed 70 iterations, instead of assessing convergence. This approach allows for a more effective evaluation of DRSGA's performance. In reality, the convergence time of the algorithm is shorter, and adjusting the relevant parameters (or establishing suitable stopping conditions) can further decrease actual running time. The algorithm running time is correlated with P and T. Generally, increasing P tends to enhance the algorithm practical performance but can also lead to an increase in execution time. The relationship between the runtime and accuracy under given values of P and T can be observed through simulations.

6. Simulations

In this section, we provide an evaluation of the accuracy and efficiency of DRSGA and compare it against the established MILP algorithm described in [21]. Although the results of MILP are accurate, we choose it for comparison since, to the best of our knowledge, it offers the fastest method for determining the exact values of r and s. Due to the limitations imposed by the acceptable running time of MILP for arbitrary digraph sizes, our comparative experiments are constrained within this limitation. It is worth emphasizing that as the digraph size increases, DRSGA can still execute and provide results. The comparative experimental outcomes suggest that DRSGA possesses a substantial advantage in terms of both time and accuracy. We employ Matlab R2021a to conduct these computational simulations. The hardware configuration consists of Intel Core i7-12650H CPUs (2.3 GHz) supporting 16 threads. Our data sample contains two parts: (1) digraphs with 15–39 nodes and 15–25 nodes, including r-robustness, (r, s)-robustness labels generated by MILP, and Erdös–Rényi random graphs (E–R graphs) [33] created by

Algorithm 2 DetermineFmax

$r \leftarrow r_{max}(\mathcal{D})$ from DRSGA in Theorem 3.1
while $r > 0$ **do**
  $s \leftarrow s_{max}(r)$ from DRSGA in Theorem 4.1
  if $s \geqslant r$ **then**
    $F_{max} \leftarrow (r-1)$
    return F_{max}
  else
    $r \leftarrow (r-1)$
$F_{max} \leftarrow 0$
return F_{max}

different p values (0.5 and 0.8); (2) graphs with 15–20 nodes, featuring $(F+1,\ F+1)$-robustness labels produced by MILP. MILP is unsuitable for solving precise robustness for larger digraphs as it entails intolerable run times. The DRSGA parameters are as follows: population number when approximately calculating $r_{max}(\mathcal{D})$: $P = 200$; others: $P = 300$; crossover rate $P_c = 0.8$; mutation rate $P_m = 0.15$; number of iterations $T = 70$ for solving r and $T = 100$ for resolving s.

Two sets of experiments tested the algorithm. The first set aimed at evaluating DRSGA's accuracy by approximately calculating ($r_{max}(\mathcal{D})$, $s_{max}(r)$) for given r and $(F_{max}+1,\ F_{max}+1)$-robustness values with DRSGA. The results appear in Fig. 4. According to Theorem 5.1, the DRSGA running result R_1 (approximate) and MILP running result R_2 (exact) have a difference of $\Delta = R_1 - R_2 \geq 0$. Consequently, we use Δ and $\alpha = \frac{R_1-R_2}{R_2}$ (or $\alpha = \frac{R_1-R_2}{R_2+1}$ when solving F_{max}) as criteria to gauge DRSGA's accuracy. For approximately calculating $r_{max}(\mathcal{D})$ and $s_{max}(r)$ values for a given r, we tested two E–R random graph types with $p = 0.5$ and $p = 0.8$ for $\mathcal{D}$, and n values ranging from 15 to 25. To ascertain $(F_{max}+1,\ F_{max}+1)$-robustness, we tested the same graph types for $\mathcal{D}$, with n values spanning from 15 to 20. This is attributed to MILP's unbearable running time when solving graphs F_{max} with over 20 nodes. Examining the experimental data reveals that errors in most graphs are small, with only a few slightly higher errors. Among 100 random graphs, only 2–5 typically exhibit errors significantly greater than the average.

The second experiment aims to evaluate the efficiency of the algorithm by comparing the runtime of MILP and DRSGA, with the results shown in Fig. 5. Both algorithms independently solved the same objective we set. DRSGA takes longer than MILP to solve less complex tasks due to its decoding process involving fitness computation. However, as the number of vertices gradually increases, particularly in solving digraphs with numerous edges, DRSGA's efficiency advantage – stemming from its polynomial time algorithm nature – becomes increasingly prominent. Specifically, in the process of approximately determining F_{max} it is often necessary to calculate multiple $s_{\max}(r)$ values for different r. When $r \neq r_{\max}(\mathcal{D})$, MILP encounters challenges in pruning, leading to substantial time overhead. Once the vertex count exceeds 20, the solution time becomes intolerable. Additionally, while the simulation terminated DRSGA's run after completing 70 iterations, the convergence time is shorter than the total running time. As the size of the digraph increases, adjusting the parameters P and T in DRSGA accordingly is recommended. The optimal values for P and T should be iteratively fine-tuned for optimal performance.

7. Conclusion

In this paper, we have considered the problem of determining r- and (r,s)-robustness for multiagent networks. To address such a problem, we first transformed this problem into minimizing a discrete function with n-elements, and then designed a heuristic algorithm to solve the transformed minimum value problem. Future work will focus on further refining and seeking improved heuristic algorithms for large-scale multiagent networks and utilizing parallel multi-threaded heuristic algorithms to further reduce computation time and enhance accuracy, while also considering the design of more effective heuristic algorithms tailored to different types of graphs.

CRediT authorship contribution statement

Jie Jiang: Writing – original draft. **Yiming Wu:** Writing – review & editing, Supervision, Methodology. **Zhaoming Zhang:** Validation, Formal analysis. **Ning Zheng:** Supervision, Methodology. **Wei Meng:** Funding acquisition, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Nos. U21A20476, 62073109) and the Zhejiang Province Public Welfare Technology Application Research Project (No. LGF21F020011).

References

[1] B. Ning, Q. Han, Z. Zuo, Distributed optimization for multiagent systems: An edge-based fixed-time consensus approach, IEEE Trans. Cybern. 49 (1) (2019) 122–132.

[2] Z. Feng, G. Hu, Y. Sun, J. Soon, An overview of collaborative robotic manipulation in multi-robot systems, Annu. Rev. Control 49 (2020) 113–127.

[3] J. He, J. Chen, P. Cheng, X. Cao, Secure time synchronization in wireless sensor networks: A maximum consensus-based approach, IEEE Trans. Parallel Distrib. Syst. 25 (4) (2014) 1055–1065.

[4] Y. Luo, Z. Wang, Y. Chen, X. Yi, H_∞ state estimation for coupled stochastic complex networks with periodical communication protocol and intermittent nonlinearity switching, IEEE Trans. Netw. Sci. Eng. 8 (2) (2021) 1414–1425.

[5] C. Zhao, J. Chen, J. He, P. Cheng, Privacy-preserving consensus-based energy management in smart grids, IEEE Trans. Signal Process. 66 (23) (2018) 6162–6176.

[6] X. Lu, Y. Jia, Scaled event-triggered resilient consensus control of continuous-time multi-agent systems under Byzantine agents, IEEE Trans. Netw. Sci. Eng. 10 (2) (2023) 1157–1174.

[7] Y. Shang, Resilient consensus for expressed and private opinions, IEEE Trans. Cybern. 51 (1) (2021) 318–331.

[8] S.M. Dibaji, H. Ishii, R. Tempo, Resilient randomized quantized consensus, IEEE Trans. Autom. Control 63 (8) (2017) 2508–2522.

[9] W. He, Z. Mo, Q.L. Han, F. Qian, Secure impulsive synchronization in Lipschitz-type multi-agent systems subject to deception attacks, IEEE/CAA J. Autom. Sin. 7 (5) (2020) 1326–1334.

[10] J. Hou, Z. Chen, M. Zhang, X. Wang, Multi-armed bandit based distributed resilient consensus and its applications in social networks, J. Franklin Inst. 359 (10) (2022) 4997–5013.

[11] H.J. LeBlanc, H. Zhang, S. Sundaram, X. Koutsoukos, Resilient continuous-time consensus in fractional robust networks, in: Proceedings of the 2013 American Control Conference, IEEE, 2013, pp. 1237–1242.

[12] H.J. LeBlanc, H. Zhang, X. Koutsoukos, S. Sundaram, Resilient asymptotic consensus in robust networks, IEEE J. Sel. Areas Commun. 31 (4) (2013) 766–781.

[13] H. Zhang, S. Sundaram, Robustness of information diffusion algorithms to locally bounded adversaries, in: Proceedings of the 2012 American Control Conference, ACC, IEEE, 2012, pp. 5855–5861.

[14] L. Tseng, N. Vaidya, Iterative approximate Byzantine consensus under a generalized fault model, in: Proceedings of the 14th International Conference on Distributed Computing and Networking, Springer, 2013, pp. 72–86.

[15] N.H. Vaidya, V.K. Garg, Byzantine vector consensus in complete graphs, in: Proceedings of the 2013 ACM Symposium on Principles of Distributed Computing, 2013, pp. 65–73.

[16] N.H. Vaidya, L. Tseng, G. Liang, Iterative approximate Byzantine consensus in arbitrary directed graphs, in: Proceedings of the 2012 ACM Symposium on Principles of Distributed Computing, 2012, pp. 365–374.
[17] D. Saldana, A. Prorok, S. Sundaram, M.F. Campos, V. Kumar, Resilient consensus for time-varying networks of dynamic agents, in: Proceedings of the 2017 American Control Conference, ACC, IEEE, 2017, pp. 252–258.
[18] S.M. Dibaji, H. Ishii, Resilient consensus of second-order agent networks: Asynchronous update rules with delays, Automatica 81 (2017) 123–132.
[19] H. Zhang, E. Fata, S. Sundaram, A notion of robustness in complex networks, IEEE Trans. Control Netw. Syst. 2 (3) (2015) 310–320.
[20] H.J. LeBlanc, X.D. Koutsoukos, Algorithms for determining network robustness, in: Proceedings of the 2nd ACM International Conference on High Confidence Networked Systems, 2013, pp. 57–64.
[21] J. Usevitch, D. Panagou, Determining r-and (r,s)-robustness of digraphs using mixed integer linear programming, Automatica 111 (2020) 108586.
[22] L. Guerrero-Bonilla, A. Prorok, V. Kumar, Formations for resilient robot teams, IEEE Robot. Autom. Lett. 2 (2) (2017) 841–848.
[23] E.M. Shahrivar, M. Pirani, S. Sundaram, Robustness and algebraic connectivity of random interdependent networks, IFAC-PapersOnLine 48 (22) (2015) 252–257.
[24] L. Guerrero-Bonilla, D. Saldana, V. Kumar, Design guarantees for resilient robot formations on lattices, IEEE Robot. Autom. Lett. 4 (1) (2018) 89–96.
[25] D. Saldaña, L. Guerrero-Bonilla, V. Kumar, Resilient backbones in hexagonal robot formations, in: Proceedings of the 14th International Symposium on Distributed Autonomous Robotic Systems, Springer, 2019, pp. 427–440.
[26] J. Usevitch, D. Panagou, r-Robustness and (r,s)-robustness of Circulant Graphs, in: Proceedings of the 56th Annual Conference on Decision and Control, CDC, IEEE, 2017, pp. 4416–4421.
[27] G. Wang, M. Xu, Y. Wu, N. Zheng, J. Xu, T. Qiao, Using machine learning for determining network robustness of multi-agent systems under attacks, in: Proceedings of the 15th Pacific Rim International Conference on Artificial Intelligence, Springer, 2018, pp. 491–498.
[28] Y. Yi, Y. Wang, X. He, S. Patterson, K.H. Johansson, A sample-based algorithm for approximately testing r-robustness of a digraph, in: Proceedings of the 2022 IEEE Conference on Decision and Control, CDC, IEEE, 2022, pp. 6478–6483.
[29] I.H. Osman, G. Laporte, Metaheuristics: A bibliography, Ann. Oper. Res. 63 (1996) 511–623.
[30] H.J. LeBlanc, X. Koutsoukos, Resilient first-order consensus and weakly stable, higher order synchronization of continuous-time networked multiagent systems, IEEE Trans. Control Netw. Syst. 5 (3) (2017) 1219–1231.
[31] S. Mirjalili, S. Mirjalili, Genetic algorithm, Evol. Algorithms Neural Netw.: Theory Appl. (2019) 43–55.
[32] J. Opacic, A heuristic method for finding most extrema of a nonlinear functional, IEEE Trans. Syst. Man Cybern. (1) (1973) 102–107.
[33] S. Janson, A. Rucinski, T. Luczak, Random Graphs, John Wiley & Sons, 2011.

Jie Jiang currently working toward the M.S. degree in cyberspace security with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His main research interests include resilient consensus control, distributed optimization and distributed system security.

Yiming Wu received the B.Eng. degree in automation and the Ph.D. degree in control science and engineering from Zhejiang University of Technology, Hangzhou, China, in 2010 and 2016, respectively. He held a visiting position from 2012 to 2014 with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. Since July 2016, he has been with the Hangzhou Dianzi University, Hangzhou, China, where he is currently an Associate Professor with the School of Cyberspace. His main research interests include multiagent systems, security and privacy theory, iterative learning control, and applications in intelligent transportation systems and sensor networks.

Zhaoming Zhang currently working toward the M.S. degree in cyberspace security with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests lie primarily in multi-agent systems and formation control.

Ning Zheng received the M.S. degree in computer application from Zhejiang University, Hangzhou, China, in 1990. He is currently a Full Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include multiagent security, information management systems, and privacy preservation.

Wei Meng received the B.E. and M.E. degrees from Northeastern University, Shenyang, China, in 2006 and 2008, respectively, and the Ph.D. degree in control and instrumentation from the Nanyang Technological University, Singapore, in 2013.

From 2012 to 2017, he was a Research Scientist with the UAV Research Group, Temasek Laboratories, National University of Singapore, Singapore. He is currently with the School of Automation, Guangdong University of Technology, Guangdong, China, as a Professor. His current research interests include unmanned systems, cooperative control, multirobot systems, and localization and tracking.