Iterative Role Negotiation via the Bilevel GRA++ With Decision Tolerance

Qian Jiang, Dongning Liu, *Senior Member, IEEE*, Haibin Zhu, *Senior Member, IEEE*, Shijue Wu, Naiqi Wu, *Fellow, IEEE*, Xin Luo, *Senior Member, IEEE*, and Yan Qiao, *Senior Member, IEEE*

***Abstract*—Role negotiation (RN) is situated at the initial stage of the role-based collaboration (RBC) methodology and is independent of the subsequent agent evaluation and role assignment (RA) processes. RN is to determine the roles and the resource requirements for each role. In existing RBC-related research, RN is assumed to be static. This means that the roles and the resource requirements for each role are predetermined by decision-makers. However, the resources allocated to each role can vary. At this time, iterative RN outcomes will have different RA results. There may not be a direct dominant relationship between different RA outcomes, especially when solving group role assignment (GRA) with multiple objectives (GRA++) problems, which makes it even more complex. To address these concerns, we introduce the original bilevel GRA++ (BGRA++) model. Specifically, at the lower level of BGRA++, a strategy is designed for quantifying iterative RNs. For the upper level, we introduce the novel GRA-NSGA-II algorithm for the RA process. Finally, we introduce the concept of decision tolerance to assist decision-makers in selecting the optimal solution from the multiple RNs. Last, simulation experiments are conducted to verify the robustness and practicability of the proposed method. Comparisons and discussions show that the proposed solution is highly competitive for solving the GRA++ problem with iterative RN.**

***Index Terms*—Bilevel GRA++ (BGRA++), decision tolerance, group role assignment (GRA), GRA with multiple objectives (GRA++), GRA-NSGA-II algorithm, iterative role negotiation (RN).**

Manuscript received 11 January 2024; revised 12 May 2024; accepted 3 June 2024. Date of publication 26 June 2024; date of current version 3 December 2024. This work was supported in part by the National Natural Science Foundation of China under Grant 62072120, in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under Grant RGPIN2018-04818, in part by the Social Sciences and Humanities Research Council of Canada (SSHRC) Insight under Grant IG 435-2023-1056, and in part by the Science and Technology development fund (FDCT), Macau SAR under Grant 0018/2021/A1 and Grant 0083/2021/A2. *(Corresponding author: Yan Qiao.)*

Qian Jiang, Naiqi Wu, and Yan Qiao are with Macao Institute of Systems Engineering and Collaborative Laboratory for Intelligent Science and Systems, Macau University of Science and Technology, Macao 999078, China (e-mail: qjiang.ieee@gmail.com; nqwu@must.edu.mo; yqiao@must.edu.mo).

Dongning Liu is with the School of Computer Science and Technology, Guangdong University of Technology, Guangdong, Guangzhou 510006, China, and also with the Faculty of Data Science, City University of Macau, Macao 999078, China (e-mail: liudn@gdut.edu.cn).

Haibin Zhu is with the Collaborative Systems Laboratory (CoSys Laboratory), Nipissing University, North Bay, ON P1B 8L7, Canada (e-mail: haibinz@nipissingu.ca).

Shijue Wu is with Department of Science and Technology of Guangdong Province, Guangdong Science & Technology Infrastructure Center, Guangzhou, Guangdong 510033, China (e-mail: wusj@gdcc.com.cn).

Xin Luo is with the College of Computer and Information Science, Southwest University, Chongqing 400715, China (e-mail: luoxin21@gmail.com).

Digital Object Identifier 10.1109/TCSS.2024.3409893

Nomenclature

$\mathcal{R}$	Set of roles.
$\mathcal{A}$	Set of agents.
L	Role range vector.
m	Number of agents.
n	Number of roles.
n_a	Number of required agents for roles
Q	Qualification matrix.
T	Role assignment matrix.
σ	Group performance.
Q_v	Multiobjective qualification matrix vector.
K	Number of objectives.
k	Current index of objectives.
$\mathcal{W}$	Decision preference vector.
U	Upper role range vector.
$\mathfrak{L}$	Candidate set about role range vector.
σ_k	Group performance under multiple objectives.
σ^p	Group performance with decision preference.
Π	Set-valued mapping.
α	Decision tolerance threshold.
Φ	Pareto solution set.
P_0	Initial parent population for each vector L in Π.
t	Number of iterations.
P_t	Final population after t rounds of iterations.
RBC	Role-based collaboration.
RN	Role negotiation.
AE	Agent evaluation.
RA	Role assignment.
GRA	Group role assignment.
GRA++	GRA with multiple objectives.
WS-GRAP	Weighted-sum GRA with preferences.
FWS-GRAP	Flexible WS-GRAP.
BGRA++	Bilevel GRA++.
TAP	Team assignment problem
PS	Pareto Set.
PF	Pareto Front.

I. Introduction

Role-based collaboration (RBC) [1], [2], [3], [4], [5] a burgeoning cybernetics methodology, is designed to utilize roles as the foundational mechanism for optimizing collaborative activities [6], [7], [8], [9], assigning the right roles to the right agents in a cooperative system. Due to its inherent interpretability, it has developed into a new problem-solving paradigm to solve industry or engineering problems [2], [10], [11], [12], [13], [14], [15], [16], [17], e.g., team assignment problem (TAP) [2], [12], [13], [14], deployment of multi-SUAV systems [10], cloud computing problem [11], spatiotemporal crowdsourcing problem (SCP) [15], [17], and cold storage defrosting problem [16].

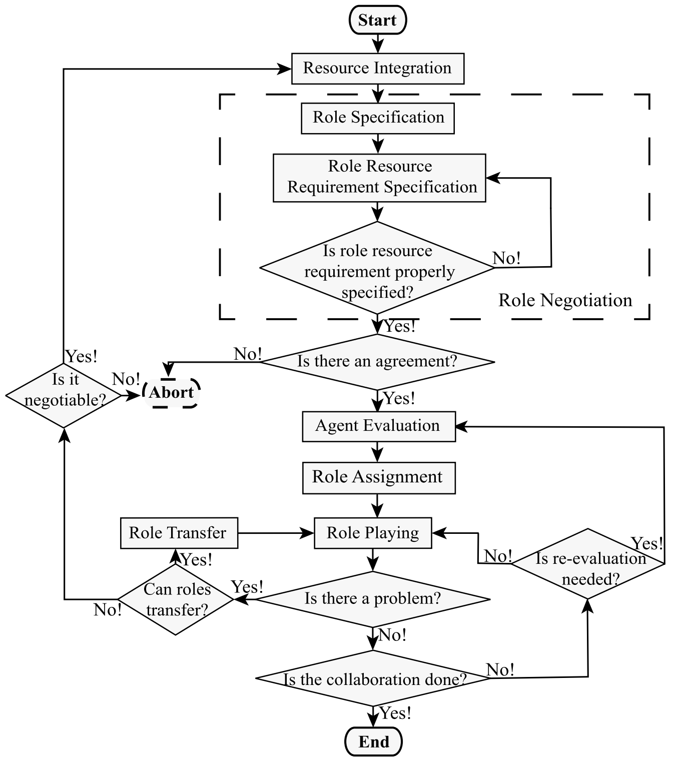


Fig. 1. RBC process in the latest related research [1], [2], [3], [4], [5], [14], [15], [16], [17].

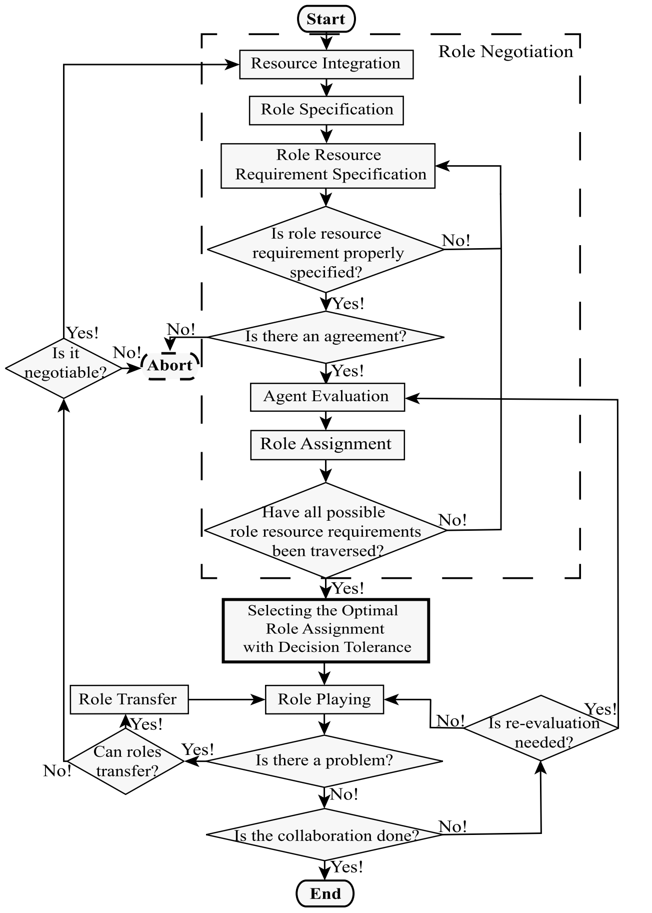


Fig. 2. Revised process of RBC in this article. Note: The dashed black boxes and the solid black box are the contributions of this work.

As delineated in [13], the foundational RBC methodology is structured around five distinct yet interrelated processes: role negotiation (RN), agent evaluation (AE), role assignment (RA), role-playing, and role transfer. The crux of RBC lies in the first three processes. Specifically, RN [17], [18], [19] encompasses three procedures: (agent) resource integration; the decomposition of tasks into their most granular, executable units, known as roles; and the determination of resource requirements and budgets for each role.

After RN, AE [10], [20] is to evaluate the qualification of agents (i.e., task executors) to play roles. Relying on the results of this AE, the RA process subsequently employs its group role assignment (GRA) with constraints (GRA+) or GRA with multiple objectives (GRA++) models [1], [2], [21] to assign the optimal agents to roles. Eventually, role-playing [1] and role transfer [22], [23] are stages that occur postassignment and involve execution and dynamic adjustments.

Up to now, existing RBC-related research [1], [2], [3], [4], [5], [10] [14], [15], [16], [17] assumes that RN is a static process, as illustrated in Fig. 1. That is, the resources of agents, the roles, and the required number of agents (i.e., task executors) for each role (i.e., task) are predetermined by decision-makers or subject-matter experts. It is easy to understand that in stable systems, both agent resources and roles are relatively stable and determined. For instance, in solving the TAP with the GRA++ model, Liu et al. [2], [3] predetermine the required number of candidates (agents) for each position (role) based on expert knowledge, such as HR's needs. In addressing the relay communication issues in multi-UAV system deployments, Jiang et al. [10] employ a role awareness algorithm to ascertain the number of drones (agents) needed from UAV bases (agents) for signal relay nodes (roles). Based on this, they design the revised GRA+ model to solve the deployment problem of multi-UAV systems. In the task assignment problem on crowdsourcing platforms, Liang et al. [15] design the corresponding GRA+ model and its corresponding task assignment algorithm based on the premise of determining the necessary workers (agents) for tasks (roles). In the multievaporator defrosting issue in cold storage, Wen et al. [16], upon establishing the required number of defrosting cycles (agents) for each evaporator (role), propose the modified GRA+ model and its scheduling algorithm.

However, allocated agent resources for each role may be uncertain. Taking the example of a company assembling a temporary team for a new project: when the new project is overwhelmingly large and there is a pressing timeline, coupled with a lack of specific expertise, the HR department might only be able to provide an overall recruitment headcount for the candidates (i.e., agents), rather than detailed headcounts for each specific position (i.e., role) within the team. That indicates RN is an iterative process.

Once RN becomes iterative, RN, AE, and RA are no longer separated processes but are integrated, as illustrated in Fig. 2. This is because different RN outcomes correspond to different

RA results, and these results may not have a dominating relationship. Especially when solving GRA++ problems, different RA result outcomes correspond to Pareto fronts with nondominating relationships. Therefore, it is necessary to consolidate the results corresponding to all RA results to obtain the optimal solution.

To tackle the challenges, we propose the original bilevel GRA++ (BGRA++) model with decision tolerance. In the lower level of this model, our primary objective is to determine the scope of iterative RN, namely, the combinations of the number of agents allocated to roles. A brute-force method of exhaustive search could be employed, but this becomes infeasible as the scale of available agent resources increases, leading to a combinatorial explosion. To address this, we integrate the concept of Pareto optimality with GRA to calculate the Pareto set concerning the combinations of agent numbers allocated to roles. This efficient approach narrows down the search space while still considering optimal and near-optimal solutions.

Moving to the upper level of BGRA++, we introduce the GRA-NSGA-II algorithm that incorporates GRA to regulate the evolutionary process of NSGA-II. This integration ensures that the offspring generated follows the specified constraints in BGRA++. With the GRA-NSGA-II algorithm, we can obtain multiple Pareto fronts based on the Pareto set acquired from the lower level. Then, we utilize the concept of decision tolerance to select the proper solution for decision-makers. A real-world scenario and a suite of simulated comparative experiments are conducted to validate the proposed method.

Please note that we opt for the NSGA-II algorithm here because it is a widely used classic approach [24] for solving multiobjective optimization problems. Our primary focus is not on designing brand-new multiobjective optimization algorithms but rather on naturally integrating the GRA model with these optimization algorithms to effectively govern its evolution, thereby efficiently solving GRA-related problems. The NSGA-II algorithm utilized here can be substituted with other multiobjective optimization algorithms, such as MOPSO [25] and NSGA-III [26], as appropriate.

In summary, the contributions of this article are listed as follows: 1) this article formalizes iterative RN in RBC via the original BGRA++ model with decision tolerance model; 2) in the lower level of BGRA++, the concept of Pareto optimality is incorporated with GRA to compute the Pareto set regarding resource assignment for roles; 3) for the upper level of the BGRA++, the original GRA-NSGA-II algorithm cooperates with the concept of decision tolerance to solve the BGRA++; and 4) the proposed method makes RBC more versatile and adaptable to various engineering problems.

The structure of this article is arranged as follows. We describe a real-world scenario related to the proposed problem in Section II. Thereafter, in Section III, we formalize the problem with the original BGRA++ model. Next, we illustrate pertinent algorithms designed for the BGRA++ model in Section IV. Then, in Section V, we conduct a suite of simulation-based comparative experiments. Last, we introduce related research and efforts in Section VI and draw some conclusions in Section VII.

TABLE I
REQUIRED POSITIONS AND QUALITIES

Position	Evaluation Criteria[a]				
	(a)	(b)	(c)	(d)	(e)
Algorithm Engineer	0.10	0.95	0.05	0.18	0.41
Front-end Engineer	0.16	0.52	0.21	0.02	0.50
Back-end Engineer	0.80	0.64	0.68	0.01	0.58
Tester	0.55	0.89	0.36	0.87	0.60

II. REAL-WORLD SCENARIO

Establishing new business groups is a crucial aspect of thriving companies. As an example, an information technology (IT) company may create an autonomous driving system group in response to the burgeoning new energy vehicle market. Two decision-makers are vital in the process of establishing a new business group: the Chief Technology Officer (CTO) Bob and the Human Resources Director (HRD) Ann.

First, Ann has recorded that 50 candidates are applying for the new autonomous driving system group in company X. Then, Bob compiles a list of the positions and qualities required for the new business group, as shown in Table I. Additionally, Bob organizes systematic interviews and computer tests for all candidates, eventually presenting the scores in Table II. Thereafter, Ann, considering the company's budget, determines the headcount (e.g., 25 people) for the new business group, collects the CTO's preferences for required qualities by position (i.e., Table III), and gathers the candidates' preferences for positions (i.e., Table IV). She must strike a balance between group performance and the preferences of both the CTO and the candidates, recognizing that both factors are equally important.

Following deliberations with Bob, Ann conducts a quantitative appraisal of the candidate pool, evaluating the collective efficacy of potential hires. This evaluation incorporates preferences from both the CTO and the candidates themselves, encapsulated in (1) and (2), which delineate the group performance metrics from dual perspectives. Note that i is the index of candidates, j is the index of positions, x is the index of evaluation criteria (i.e., x $\{0, 1, 2, 3, 4\}$), $\mathcal{T}_1$ and $\mathcal{T}_2$ record the values in Tables I and II, respectively, and $\mathcal{P}_1$ and $\mathcal{P}_2$ record the values in Tables III and IV

$$Q_1[i,j] = \sum_{x=0}^{4}(\mathcal{T}_2[i,x] - \mathcal{T}_1[j,x]) \tag{1}$$

$$Q_2[i,j] = \left(\sum_{x=0}^{4}((\mathcal{T}_2[i,x] - \mathcal{T}_1[j,x]) \times \mathcal{P}_1[j,x])\right) \times \mathcal{P}_2[i,x]. \tag{2}$$

Based on the business group establishment problem described above, we are dealing with a one-to-many (1-M) assignment issue. Namely, one candidate can only work in one position at a given time, but multiple candidates can assume roles in the same position simultaneously.

Furthermore, Table I outlines the threshold levels of skills required for various positions, while Q_1 represents the current performance capabilities of the group (i.e., team). In the context

TABLE II
EVALUATION SCORES OF CANDIDATES

Candidate[a]	Evaluation Criteria[b]				
	(a)	(b)	(c)	(d)	(e)
Candidate 1	0.60	0.43	0.04	0.88	0.98
Candidate 2	0.90	0.00	0.63	0.87	0.44
Candidate 3	0.68	0.04	0.75	0.10	0.04
...	...	...	...	...	...
Candidate 48	0.54	0.09	0.93	0.95	0.01
Candidate 49	0.45	0.51	0.02	0.87	0.46
Candidate 50	0.96	0.01	0.14	0.67	0.96

[a]For display purposes, only the first and last three rows are shown. To eliminate the dimensional difference of different criteria, these criteria are quantized as values between 0 and 1. [b](a) Candidate's degree; (b) grade point average (GPA); (c) expertise; (d) working experience; and (e) communication skill.

TABLE III
PREFERENCES OF THE CTO FOR POSITION CRITERIA

Position	Evaluation Criteria[a]				
	(a)	(b)	(c)	(d)	(e)
Algorithm Engineer	0.85	0.99	0.27	0.45	0.93
Front-end Engineer	0.79	0.06	0.41	0.39	0.11
Back-end Engineer	0.35	0.91	0.00	0.33	0.65
Tester	0.51	0.88	0.15	0.07	0.99

[a](a) Candidate's degree; (b) grade point average (GPA); (c) expertise; (d) working experience; and (e) communication skill.

TABLE IV
POSITION PREFERENCE OF CANDIDATES

Candidate	Position[a]			
	(1)	(2)	(3)	(4)
Candidate 1	1.00	0.90	0.41	0.54
Candidate 2	1.00	0.74	0.06	0.33
Candidate 3	0.33	1.00	0.56	0.22
...	...	...	...	...
Candidate 48	0.06	1.00	0.89	0.21
Candidate 49	0.37	0.05	1.00	0.62
Candidate 50	0.80	0.91	1.00	0.16

[a](1) Algorithm engineer; (2) front-end engineer (GPA); (3) back-end engineer; and (4) tester. Candidates' preferences for positions are quantized as values between 0 and 1.

of the TAP, building a sustainable team [2], [27] requires considering not only the group performance but also the preferences of the CTO and those of the potential candidates, as illustrated by Q_2. That is, decision-makers need to consider two objectives in this scenario: 1) maximizing the group performance of candidates; and 2) maximizing the group performance considering both decision-makers' preferences and candidates' preferences. As such, this problem can be formalized as a GRA++ problem.

However, what distinguishes it from existing GRA++ problems [1], [2], [21] is the specificity of assignment requirements. Here, we are only given a total number of candidates required for all positions—25 in this scenario—but the exact number of candidates needed for each position is uncertain. Existing GRA++ models operate under the assumption that the number of candidates needed for each role is predetermined. When the required number of candidates for each position becomes uncertain, these conventional GRA++ models become inapplicable.

To address this, we introduce the BGRA++ model with decision tolerance. In the following section, we will provide a formal definition of the proposed model based on the real-world scenario described.

III. BGRA++

In this section, we provide a comprehensive description of the proposed BGRA++ model for solving the TAP mentioned in Section II. First, we offer the optimistic formulation of the BGRA++ model. Subsequently, we discuss the specific details and associated strategies of the BGRA++ model for both the lower and upper levels.

A. BGRA++: The Optimistic Formulation

In this section, we primarily discuss the general formalized form for the BGRA++ model. This model is based on RBC methodology and represents a significant expansion of the Environments–Classes, Agents, Roles, Groups, and Objects (E-CARGO) metamodel [1], [2], [3], [4], [5], designed to accommodate more complex industrial application problems.

The priority of utilizing the BGRA++ model is to determine the roles (i.e., tasks) and the agents (i.e., task executors). To clarify the definition of BGRA++, we use nonnegative integers m ($= |\mathcal{A}|$, where $|\mathcal{A}|$ is the cardinality of set $\mathcal{A}$) to express the size of the agent set $\mathcal{A}$, n ($= |\mathcal{R}|$) the size of the role set $\mathcal{R}$. Here, $i \in A = \{0, 1, \ldots, m-1\}$ and $j \in R = \{0, 1, \ldots, n-1\}$ represent the indices of agents and roles, respectively. The critical definitions of BGRA++ are as follows. To clarify the new contributions, all the definitions without citations are proposed in this work for the first time.

Definition 1 [Role]: A role [2], [28] is defined as $r \overset{\text{def}}{=} <id, ®>$.

Note: *id* denotes the identifier of *r*, e.g., *role j* ($j \in R$); ® represents the set of requirements or properties for agents to perform *r*. In other words, ® specifies the constraints for the agents to play the roles. In the real-world scenario, roles are positions in the new business group, and $n = 4$.

Definition 2 [Agent]: An agent [1], [29] is defined as $a \overset{\text{def}}{=} <id, @>$.

Note: *id* is the identification of a, e.g., *agent i* ($i \in A$); @ is the set of *a*'s values corresponding to the abilities required in the group @ expresses the agents' performances about ®. The specific definition of @ is illustrated in Definition 5. In the real-world scenario, agents are the candidates, and $m = 50$.

Definition 3 [Role Range Vector]: A *role range vector* [4] *L* is an *n*-dimensional vector of the number of agents required for roles, i.e., $L[j] \in \mathbb{N}^+$.

Note: Essentially, the symbol *L* denotes the resource assignment for each role. Previous GRA-related models [2], [14], [29], [30], [31] typically assume that *L* can be determined beforehand, using expert knowledge or insights from decision-makers. In the real-world scenario, vector L is uncertain.

Definition 4 [Required Number of Agents] [1]*:* The symbol n_a denotes the number of required agents for roles. That is, $n_a = \sum_{j=0}^{n-1} L[j]$.

Note: The physical significance of n_a represents the overall available agent resources. In the real-world scenario, $n_a = \sum_{j=0}^{3} L[j] = 50$.

Definition 5 [Qualification Matrix]: A *qualification matrix* [20] Q is an $m \times n$ matrix, where $Q[i, j] \in \mathbb{R}^+$ expresses the suitability of an agent i for a role j.

Note: The physical meaning of Q represents the weights of the control variables.

Definition 6 [RA Matrix]: A *RA matrix* [2], [14], [29], [30], [31], [32] T is defined as an $m \times n$ control variable matrix, where $T[i, j] \in \{0, 1\}$ indicates whether or not an agent i is assigned to a role j. $T[i, j] = 1$ means yes and 0 no.

Definition 7 [Group Performance]: The *group performance* [18], [33] σ is defined as the sum of the assigned agents' qualifications, i.e.,

$$\sigma(T) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} Q[i, j] \times T[i, j].$$

Note: The physical meaning of σ represents the objective function.

Definition 8 [GRA Model]: Given $\mathcal{A}(|\mathcal{A}| = m)$, $\mathcal{R}(|\mathcal{R}| = n)$, Q, and L, the condensed GRA model is to find a workable T to obtain the following:

$$\max\ \sigma(T) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} Q[i, j] \times T[i, j]$$

subject to

$$T[i, j] \in \{0, 1\}\ (i \in A, j \in R) \tag{3}$$

$$\sum_{i=0}^{m-1} T[i, j] = L[j] (j \in R) \tag{4}$$

$$\sum_{j=0}^{n-1} T[i, j] \leq 1 (i \in A) \tag{5}$$

where (3) expresses the 0–1 variable, (4) indicates the number of agents required for roles, and (5) ensures that each agent can only play one role at a time.

Note: Here, we introduce GRA, which is used in the designed algorithms of the BGRA++ model.

Extant GRA++ models [2], [14], [29], [30], [31] predominantly employ a priori methods, such as the weighted-sum (WS) approach [34], to reduce multiobjective optimization problems to their single-objective equivalents. In this context, we introduce new formal definitions to delineate the weighted-sum GRA with preferences (WS-GRAP) model, a frequently utilized variant within the GRA++ framework.

Definition 9 [Multiobjective Qualification Matrix Vector]: A *multiobjective qualification matrix vector* Q_v is a $K \times (m \times n)$ vector, where $Q_v[k, i, j] \in \mathbb{R}^+$ expresses the suitability of an agent i for a role j under the kth ($k \in \{1, 2, 3, \ldots, K\}$) objectives.

Note: In the real-world scenario, $K = 2$, and the qualification matrices Q_1 and Q_2 are formalized in (1) and (2), respectively. Thus, Q_v in the real-world scenario is $[Q_1, Q_2]$.

Definition 10 [Decision Preference Vector]: The *Decision Preference Vector* $\mathcal{W} = [w_1, w_2, \ldots, w_k]$ ($w_k \in (0, 1)$, and $\sum_{k=1}^{K} w_k = 1$) indicates the decision-makers' preference for the kth ($k \in \{1, 2, 3, \ldots, K\}$) objectives.

Note: In the real-world scenario, $\mathcal{W} = [w_1, w_2] = [0.5, 0.5]$.

Definition 11 [WS-GRAP Model]: Given $\mathcal{A}(|\mathcal{A}| = m)$, $\mathcal{R}(|\mathcal{R}| = n)$, K ($K \geq 1$), Q_v, $\mathcal{W}$, and L, the WS-GRAP model is to find a workable T to obtain the following:

$$\max\ \sigma^{\mathrm{WS-GRAP}}(T) = \sum_{k=1}^{K} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (\mathcal{W}[k] \times Q_v[k, i, j] \times T[i, j])$$

subject to (3)–(5).

As elucidated in Definition 11, WS-GRAP for solving the real-world scenario faces a key challenge. A predefined role range vector L is mandatory, implying that the number of agents needed for all roles must be established in advance. This requirement renders the WS-GRAP model ineffective in cases of uncertain L. To surmount this challenge and render the model tractable, we introduce the following novel definitions for enumerating the uncertain vector L.

Definition 12 [Upper Role Range Vector]: An *upper role range vector* [1] U is an n-dimensional vector of the number of agents allowed for roles, i.e., $U[j] \in \mathbb{N}^+$.

Note: In the real-world scenario, $U = [n_a, n_a, n_a, n_a] = [50, 50, 50, 50]$.

Definition 13 [FWS-GRAP Model]: Given $\mathcal{A}(|\mathcal{A}| = m)$, $\mathcal{R}(|\mathcal{R}| = n)$, K ($K \geq 1$), Q_v, $\mathcal{W}$, L, U, and n_a, the flexible WS-GRAP (FWS-GRAP) model is to find a workable T to obtain the following:

$$\max\ \sigma^{\mathrm{FWS-GRAP}}(T) = \sum_{k=1}^{K} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (\mathcal{W}[k] \times Q_v[k, i, j] \times T[i, j])$$

subject to (3), (5), and

$$L[j] \leq \sum_{i=0}^{m-1} T[i, j] \leq \mathrm{U}[j] (j \in R) \tag{6}$$

$$\sum_{i=0}^{m-1} \sum_{j=0}^{n-1} T[i, j] = n_a \tag{7}$$

where (6) stipulates the number of agents should be more than required and under the budgets, and (7) ensures the required number of agents.

In FWS-GRAP, we can set L as the minimum required number of agents for each role, i.e., $L = [1, 1, 1, 1]$, and enumerate all feasible L by imposing constraints defined in (6) and (7). In this case, FWS-GRAP is capable of directly solving the problem in the real-world scenario. However, it encounters two significant computational bottlenecks. First, the complexity of (6) is of the order $O(n \times (n_a)^m)$, which grows exponentially with the increase of m, easily leading to a combinatorial explosion. Second, the (F)WS-GRAP model empirically converts multiobjective optimization problems into single-objective problems via the weighted-sum method, a transformation that could compromise group performance and potentially overlook optimal solutions.

To address these bottlenecks, we devise another approach to transform L from uncertainty to determinism. To this end, the BGRA++ model with decision tolerance has been introduced along with the following new definitions.

Definition 14 [Candidate Set about Role Range Vector]: The candidate set about role range vector $\mathfrak{L}$ ($|\mathfrak{L}| \in \mathbb{N}^+$) includes all candidate vectors Ls which satisfy the constraint that $\sum_{j=0}^{n-1} L[j] = n_a$.

Note: In the real-world scenario, $\mathfrak{L}$ aggregates all feasible candidate vectors Ls which adhere to the resource constraint that $\sum_{j=0}^{n-1} L[j] = 50$.

Definition 15 [Group Performance under Multiple Objectives]: The group performance σ_k under kth objective is defined as

$$\sigma_k(L, T) = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (Q_v[k, i, j] \times T[i, j])(1 \le k \le K).$$

Note: In the real-world scenario, given L, $\sigma_1 = \sum_{i=0}^{49}\sum_{j=0}^{3} Q_1[i, j] \times T[i, j]$ and $\sigma_2 = \sum_{i=0}^{49}\sum_{j=0}^{3} Q_2[i, j] \times T[i, j]$.

Definition 16 [Multiobjective Group Performance]: The multiobjective group performance $\overline{\sigma}_{\mathrm{M}}$ is defined as

$$\overline{\sigma}_{\mathrm{M}}(L, T) = (\sigma_1(L, T), \sigma_2(L, T), \ldots, \sigma_K(L, T)). \tag{8}$$

Note: The multiobjective group performance $\overline{\sigma}_{\mathrm{M}}$ denotes the set of objective functions. Symbols L and T represent the control variables in the BGRA++ model.

Definition 17 [BGRA++ Model]: Given $\mathcal{A}(|\mathcal{A}| = m)$, $\mathcal{R}(|\mathcal{R}| = n)$, K ($K \ge 1$), and Q_v, the BGRA++ model is to find a workable T to obtain the following:

$$\max\ \overline{\sigma}_{\mathrm{M}}(L, T) = (\sigma_1(L, T), \sigma_2(L, T), \ldots, \sigma_K(L, T))$$

subject to (3)–(5), and

$$L \in \mathfrak{L} \tag{9}$$

where vector L ($L \in (\mathbb{N}^+)^n$) represents the lower level variable of the BGRA++ model, and matrix T ($T \in \{0, 1\}^{m \times n}$) represents the upper level variable.

In the formulation, two key differences distinguish the BGRA++ model (as per Definition 17) from the WS-GRAP model (as per Definition 11). First, the vector L transitions from a constant to a control variable. As a result, we need to determine the set $\mathfrak{L}$ before deriving a workable T matrix, in accordance with (4). Second, the objective function of the model fundamentally shifts from a weighted-sum single-objective to multiobjective optimization. For this concern, it is crucial to develop a new solution strategy for BGRA++ models that can harmonize group performance with decision-makers' preferences. These two disparities have not been explored in the previous GRA-related research.

Algorithm 1: Calculating the Set $\mathfrak{L}$

```
Input:
➢ The number of required agents for roles $n_a$
➢ The cardinality of roles $n$
Output:
➢ Candidate set about role range vector $\mathfrak{L}$
begin
1:   $\mathfrak{L} \leftarrow \emptyset$; /* Initialization */
2:   $\Upsilon \leftarrow \{1, 2, \ldots, n_a-1\}$;
3:   temp ← Combinations($\Upsilon$, $n-1$); /* temp is a tentative value */
4:   for item in temp do
5:       $\widehat{L} \leftarrow$ []; /* [] represents an empty vector */
6:       item ← [0, item, $n_a$];
         /* |item| represents the cardinality of the vector item */
7:       for $z \leftarrow 1$ to |item| do
8:           $\widehat{L} \leftarrow [\widehat{L}, item[z]-item[z-1]]$;
9:       end for
10:      $\mathfrak{L} \leftarrow \mathfrak{L} \cup \{\widehat{L}\}$;
11:  end for
12:  return $\mathfrak{L}$;
end
```

B. BGRA++ at the Lower Level

Section III-B is to determine the candidate set $\mathfrak{L}$ of the role range vector L, which specifies the number of agents required for each role. This process is the foundation of the BGRA++ model, as determining $\mathfrak{L}$ is essential before solving for the upper level control variable T in the BGRA++ model.

Drawing on Definitions 14 and 17, the critical part for the lower level of the BGRA++ model is to compute the set $\mathfrak{L}$. It is essentially an agent resource grouping problem. Here, we employ the balls and bars method [35] in combinatorial mathematics to help solve this issue. Consider roles are represented by discrete sections, delineated by separators (referred to as "bars"), ensuring that each role is confined to its designated area without overlap with adjacent roles. Agents as balls, with n_a denoting the total number of balls, and we propose Algorithm 1 to tackle it. Here, we supplement several additional symbols to better describe Algorithm 1.

1) Symbol $\widehat{L}$ represents the tentative value of the vector L.
2) Symbol Υ collects the possible locations of bars among n_a agents, and $\Upsilon = \{1, 2, \ldots, n_a-1\}$.
3) Function *Combinations*(Υ, $n-1$) collects the positions of $n-1$ bars among n_a agents, ingeniously corresponding to the allocation of agent resources across n roles. For instance, given $n = 3$ and $n_a = 4$, the output of *Combinations*(Υ, $n-1$) would be {[1, 2], [1, 3], [2, 3]}. In Fig. 3, the position of two bars is [1, 3].

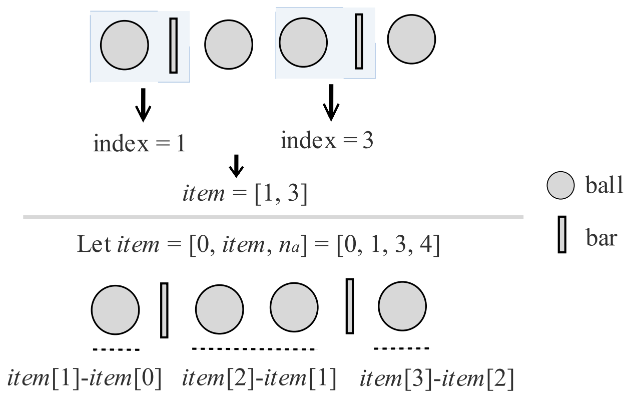


Fig. 3. Simple case to help illustrate Algorithm 1. Note: n_a represents the total number of balls. Here, we employ the balls and bars method [35] in combinatorial mathematics to assist in solving the set $\mathfrak{L}$.

Algorithm 1 depicts the pseudocode of calculating the set $\mathfrak{L}$. As the ball groupings correspond to bar positions, we aim to establish a mathematical relationship between these positions and the groupings. Consequently, the outer loop in Algorithm 1 signifies the various combinations of bar positions. Furthermore, lines 5–8 in Algorithm 1 establish the mapping relationship between the bar positions and ball groupings. Here, we also utilize a straightforward case to illustrate the details of the corresponding mapping operation (see Fig. 3).

The optimal solution for the BGRA++ model involves solving all the Pareto fronts corresponding to vector Ls in the set $\mathfrak{L}$. Based on Algorithm 1, the cardinality of the set $\mathfrak{L}$ is $\binom{m-1}{n-1} = ((m-1)!)/((n-1)! \times (m-n)!)$. As the number of roles and agents increases, the cardinality of $\mathfrak{L}$ can potentially expand dramatically when trying to solve the BGRA++ model.

To mitigate this problem, it is necessary to swiftly filter through the vectors in the set $\mathfrak{L}$, aiming to select a new set $\mathcal{H}$ that likely contains the optimal solution. We will compare the effectiveness of this strategy through extensive simulation experiments in the experimental section. Based on this requirement, we introduce a new formulation for the BGRA++ model, which provides an acceptable tradeoff between the solution time and the group performance. Here, we also supplement some new definitions for clarity.

Definition 18 [Pareto Domination]: Given L and any two vectors $u = [(\sigma_1)_u, (\sigma_2)_u, \ldots, (\sigma_K)_u]$ and $\omega = [(\sigma_1)_\omega, (\sigma_2)_\omega, \ldots, (\sigma_K)_\omega]$ in the objective space, we say that u **dominates** ω *iff* $u \neq \omega$ and $\forall k \in \{1, 2, 3, \ldots, K\}$, $(\sigma_k)_u \geq (\sigma_k)_\omega$, denoted as $u \succ \omega$.

Definition 19 [Pareto Optimal]: Given L, a RA matrix $T \in \{0,1\}^{m\times n}$ is **Pareto optimal** if there is no $T' \in \{0,1\}^{m\times n}$ $(T' \neq T)$ such that $\overline{\sigma}_M(L, T') \succ \overline{\sigma}_M(L, T)$.

Definition 20 [Pareto Set]: Given L, the Pareto set (PS) under current vector L is a set of the Pareto optimal solutions, which can be represented as $PS \overset{\text{def}}{=} \{T \in \{0,1\}^{m\times n}$ and T is Pareto optimal$\}$.

Definition 21 [Pareto Front]: Given L, the Pareto front (PF) is composed of the solutions in PS, which can be defined as $PF \overset{\text{def}}{=} \{\overline{\sigma}_M(L, T) \mid T \in PS\}$.

Definition 22 [Group Performance with Decision Preference]: The *group performance with Decision Preference* σ^p is defined as

$$\sigma^p(L, T) = \sum_{k=1}^{K}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1} \mathcal{W}[k] \times Q_v[k, i, j] \times T[i, j]. \tag{10}$$

Definition 23 [Set-valued Mapping]: Let $\Pi : \{0,1\}^{m\times n} \rightrightarrows (\mathbb{N}^+)^n$ be a set-valued mapping and

$$\Pi = \bigcup_{k=1}^{K} \big(\operatorname*{argmax}_L \{\sigma_k(L, T) \mid (3)\text{–}(5), L \in \mathfrak{L}\}\big) \cup \operatorname*{argmax}_L \{\sigma^p(L, T) \mid (3)\text{–}(5), L \in \mathfrak{L}\} \tag{11}$$

where (11) aids in selecting vector Ls that optimize every single-objective function or the decision-makers' preferences.

Note: Mapping Π encompasses the PS regarding L. Expression (11) corresponds to the solution sets pertaining to L from several extended GRA problems. These problems can be rigorously formalized using the GRA model delineated in Definition 8, and every expanded GRA issue can be resolved by employing the IBM ILOG CPLEX optimization package (CPLEX) [36] to determine the solutions, thus acquiring Π. In the real-world scenario, $\Pi = \{[1, 22, 1, 1], [2, 21, 1, 1], [3, 20, 1, 1], [4, 19, 1, 1], [5, 18, 1, 1], [6, 17, 1, 1]\}$ and $|\Pi| = 6$.

Given these definitions, the new expression of the BGRA++ model can be formulated as

$$\max \overline{\sigma}_M(L, T) = (\sigma_1(L, T), \sigma_2(L, T), \ldots, \sigma_K(L, T))$$

subject to (3)–(5), and

$$L \in \Pi. \tag{12}$$

In the given formulation, we consider both the single-objective function and the decision-makers' preferences to provide all PS options about vector L, enabling the selection of the best point based on the upper level objectives. Unlike the traditional optimization problems, the PS about vector L will generate different PFs, and there may be no dominance relationship between them. As a result, we may need to consider the decision-makers' preferences to select the optimal Pareto solution. Still, since the upper level variable T in the BGRA++ model is discrete, there may not exist an optimal solution that satisfies the decision-makers' preferences. For these concerns, we introduce a concept of decision tolerance at the upper level of the BGRA++ model.

C. BGRA++ at the Upper Level

In this section, based on the PS for L (i.e., Π) obtained from the lower layer of the BGRA++ model, at the upper level of the BGRA++ model, we need to determine how to obtain the optimal assignment solution (i.e., matrix T) that meets the decision-maker's preferences, especially in the context of TAP under discrete events.

As mentioned in Section III-B, each candidate vector L within Π generates a corresponding PF, and there may not be a dominant relationship between these PFs (refer to Fig. 4). Thereby, we rely on decision-makers' preferences are the criteria for

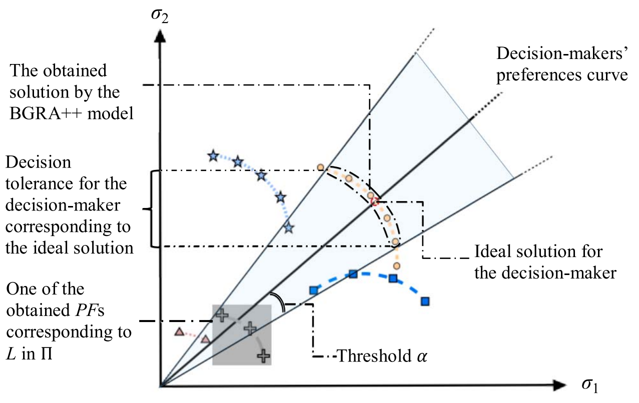

σ_2
σ_1
The obtained solution by the BGRA++ model
Decision-makers' preferences curve
Decision tolerance for the decision-maker corresponding to the ideal solution
Ideal solution for the decision-maker
One of the obtained *PF*s corresponding to *L* in Π
Threshold α

Fig. 4. Simple case to illustrate the process of the BGRA++ model at the upper level.

selecting the optimal solution of the BGRA++ problem. As an example, the bold black line in Fig. 4 expresses the decision-makers' preferences. However, since the control variables in the BGRA++ model are not continuous, an optimal solution that precisely aligns with the decision-makers' preferences may not exist. Respective of this challenge, we propose the concept of decision tolerance, with its precise definition depicted in Definition 24.

Definition 24 [Decision Tolerance Threshold]: The *decision tolerance threshold* α signifies an acceptable level of decision bias that decision-makers are willing to accommodate.

Note: The symbol α represents the range of deviation from the suboptimal solution to the optimal solution. To strike a balance between the decision-makers' preferences and the likelihood of achieving a desired solution, we employ a triangular area with the ideal optimal solution as the central axis to depict the allowable decision bias area.

Definition 25 [Pareto Solution Set]: The *Pareto solution set* Φ represents a set that contains the corresponding $\overline{\sigma}_{\mathrm{M}}$ of the Pareto solutions on the *PF*s obtained by the BGRA++ model.

Note: In Fig. 4, the set Φ collects the corresponding $\overline{\sigma}_{\mathrm{M}}$ of the Pareto points on the five *PF*s.

Here, we leverage Fig. 4 as a simple case to explain the process of the BGRA++ model at the upper level. In Fig. 4, we assume the existence of two objective functions σ_1 and σ_2 and postulate five candidate vectors Ls (i.e., $|\,\Pi\,| = 5$), corresponding to five different results of agent resource allocation for roles. The curve composed of five different types of icons in the figure represents the Pareto frontier achieved under these five results. We also assume a tolerance deviation α for the decision-maker's optimal solution; for demonstration purposes, we suppose α ranges from 0 to $\pi/2$, with the decision-maker considering both objectives equally important, i.e., $\mathcal{W} = [w_1, w_2] = [0.5, 0.5]$. Given that the TAP is a discrete event, the most satisfactory solution for the decision-maker may not exist on the *PF*s. In this case, we select the optimal solution under the decision-maker's tolerance, specifically the Pareto point on the orange circular curve that is closest to the decision-maker's preferred curve.

As illustrated in Fig. 4, since the control variable T in the BGRA++ model is discrete, no optimal solution exists that

Algorithm 2: Searching for the Solution to the BGRA++

```
Input:
➢ Pareto solution set $\Phi$
➢ Decision tolerance threshold $\alpha$
➢ Symbol $\theta$
Output:
➢ The solution to the BGRA++ $\widehat{\sigma_{\mathrm{M}}}$
Begin
1:  $\widehat{\sigma_{\mathrm{M}}} \leftarrow \emptyset$; /* Initialization */
2:  $index \leftarrow -1$; /* $index$ is a tentative value that represents the
    corresponding index about the optimal solution in $\Phi$ */
3:  $value \leftarrow -\infty$; /* $value$ is a tentative value */
4:  $\Xi \leftarrow CalculatingInclinationAngle(\Phi)$; /* $\Xi$ is a tentative
    set */
5:  for $z \leftarrow 1$ to $|\Phi|$ do
6:      $temp = \|\Phi[z]\|_2$; /* $temp$ is a tentative value, and $\|\cdot\|_2$
        represents the $\ell^2$ norm */
7:      if $value < temp$ and $\theta - \alpha \le \Xi[z] \le \theta + \alpha$ then
8:          $value \leftarrow temp$;
            $index \leftarrow z$;
9:      end if
10: end for
11: if $index \neq -1$ then
12:     $\widehat{\sigma_{\mathrm{M}}} \leftarrow \Phi[index]$;
13: end if
14: return $\widehat{\sigma_{\mathrm{M}}}$;
end
```

fully satisfies the decision-makers' preferences. In response, we utilize the threshold α to find the suboptimal solution that is acceptable to decision-makers. Algorithm 2 details the specific procedure for finding the solution to the BGRA++ model. Additionally, we introduce several supplementary symbols to enhance the description of Algorithm 2.

1) Symbol θ represents the angle of inclination about the decision-maker's preference curve. For example, in Fig. 4, θ is $\pi/4$ rad.
2) *Calculating Inclination Angle*(Φ) represents the function that can obtain the inclination angle for the pairwise elements in Φ.
3) Symbol $\widehat{\sigma_{\mathrm{M}}}$ denotes the solution derived from the BGRA++ model.

As illustrated in Algorithm 2, the Euclidean distance between the Pareto point and the origin is used as the standard to measure the quality of its solution, i.e., line 6 in Algorithm 2. This section has elaborated on the BGRA++ model. The leftover part depicts the method to solve the BGRA++ model, specifically, calculating the set Φ.

IV. GRA-NSGA-II Algorithm

In this section, we present a comprehensive explanation of how to naturally integrate genetic algorithms with the GRA model to compute the set Φ. More pertinently, we utilize the NSGA-II algorithm as an example to illustrate this combination

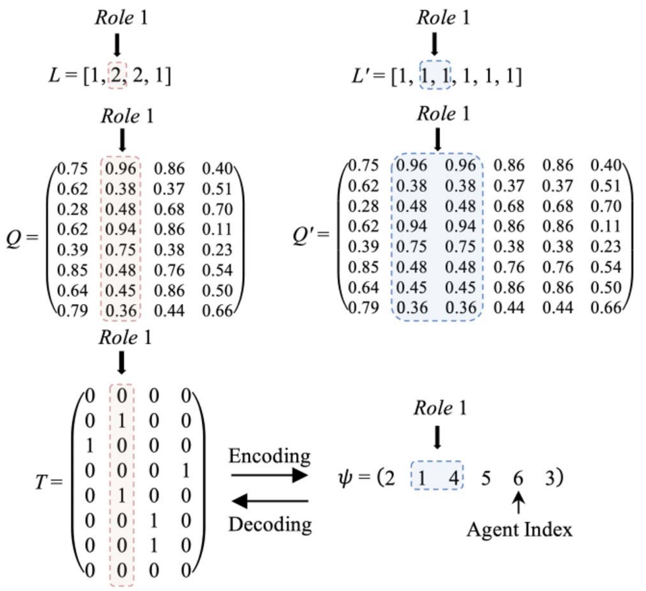


Fig. 5. Simple case to illustrate the encoding and decoding process of the GRA-NSGA-II algorithm.

because it is one of the most widely used genetic algorithms [24]. For simplicity, we refer to this modified NSGA-II algorithm as the GRA-NSGA-II algorithm. The detailed procedure of the GRA-NSGA-II algorithm is outlined as follows.

Please note that Section IV is to describe the deep integration of the GRA model with the genetic algorithm to efficiently control its evolution for better solutions instead of devising a brand-new genetic algorithm. Therefore, the NSGA-II algorithm can be replaced with other multiobjective optimization algorithms like MOPSO [25], NSGA-III [26], and others.

A. Solution Construction

Since the GRA model incorporates 0–1 constraints, as shown in (3)–(5), we employ the following well-designed encoding approach. Let $\psi = \{\psi_1, \psi_2, \ldots, \psi_t\}$ $(t\ \{1, 2, \ldots, \sum_{j=0}^{n-1} L[j]\})$ represents an encoded solution for the problem, where ψ_t is the index of the agents, i.e., $\psi_t \in A$. Essentially, ψ is a dimensionality reduction transformation of the matrix T. This transformation simplifies the integration of the GRA++ model with the NSGA-II algorithm and ensures (6) is satisfied.

Here, we utilize a simple case to explain this encoding approach. Assume that there are eight agents, four roles, and one objective function (i.e., $K = 1$, $\mathcal{W} = [1]$, and $Q_v = [Q]$). The corresponding role range vector L, normalized qualification matrix Q, and one of the candidate assignment results T are displayed on the left side of Fig. 5. To more effectively generate new solutions for the GRA-NSGA-II algorithm, we convert the 1-M assignment [4] into a one-to-one (1-1) assignment [37].

Taking *Role* 1 in Fig. 5, encircled by the dashed orange line on the left, as an example, the value of 2 in the column corresponding to *Role* 1 in L indicates that *Role* 1 requires two agents. For ease of encoding, we expand this column into adjacent columns corresponding to the numerical values. As a result of the change in the number of columns, we also need to perform a corresponding transformation on the Q matrix associated with vector L. Since the Q values for the expanded columns are identical, we accordingly expand the Q matrix for *Role 1* to form Q'. As a corollary, the control variable matrix T is expanded from a 2-D matrix into a 1-D control variable vector, ψ. The values in the vector ψ correspond to the indices of the agents. Likewise, the relationship between L and L' can be leveraged for decoding.

B. Generating New Solutions

In this section, we expound on the details of the GRA-NSGA-II algorithm's mechanisms for generating new solutions, including the procedures of crossover, mutation, and selection processes. To begin with, instead of randomly generating the initial parent population of a specific number of individuals, we utilize CPLEX to solve (13) and efficiently generate an initial parent population P_0 for each vector L in Π. This approach is adopted because the GRA model has (3)–(5), and randomly generated populations may not satisfy these constraints

$$P_0 = \underset{T}{\operatorname{argmax}}\{\sigma(L, T) \mid (3)\text{–}(5), \sigma(L, T) \in \{\sigma_1(L, T)\ \sigma_2(L, T), \ldots, \sigma_K(L, T), \sigma^p(L, T)\}\ (L \in \Pi)\}. \tag{13}$$

By using the optimal solutions by the GRA model with a single objective, we can swiftly generate a suitable P_0 that adheres to these constraints. Subsequently, we apply the encoding rules proposed in Section IV-A to transform every matrix T in P_0 into a corresponding ψ. Then, akin to the original NSGA-II algorithm, we leverage (13) to calculate the fitness of these individuals and assign each individual a nondomination rank. Based on these ranks, we can iteratively employ binary tournament selection to choose specific individuals in P_0 and generate a corresponding offspring population.

Due to the existence of (5) in GRA, the original strategy of randomly generating offspring in the NSGA-II algorithm becomes ineffective. To meet Constraint (5), it is necessary to ensure that the offspring generated by the GRA-NSGA-II algorithm do not have duplicate agent indices. Therefore, during the crossover and mutation processes of generating offspring, it is essential to control that all produced offspring satisfy the requirements of (5).

The procedure of generating the offspring population in the GRA-NSGA-II is depicted in Fig. 6. The two dashed pink squares emphasize the primary differences between the GRA-NSGA-II algorithm and the original NSGA-II during the crossover and mutation processes. After performing the crossover operation on individuals, it is crucial to check for duplicate agent indices in the newly generated individuals, as required by (5) of the GRA. For instance, in Fig. 6, individual a's offspring has a conflicting agent index 3. To resolve the conflict in the cross-matching section, we need to randomly select a replaced agent index (e.g., agent index 7) from the set of optional agent indices for individuals (i.e., $A-\psi$). Similarly, during the mutation process, we utilize each individual's set of optional agent indices to control the direction of mutation. Last, we merge the parent and

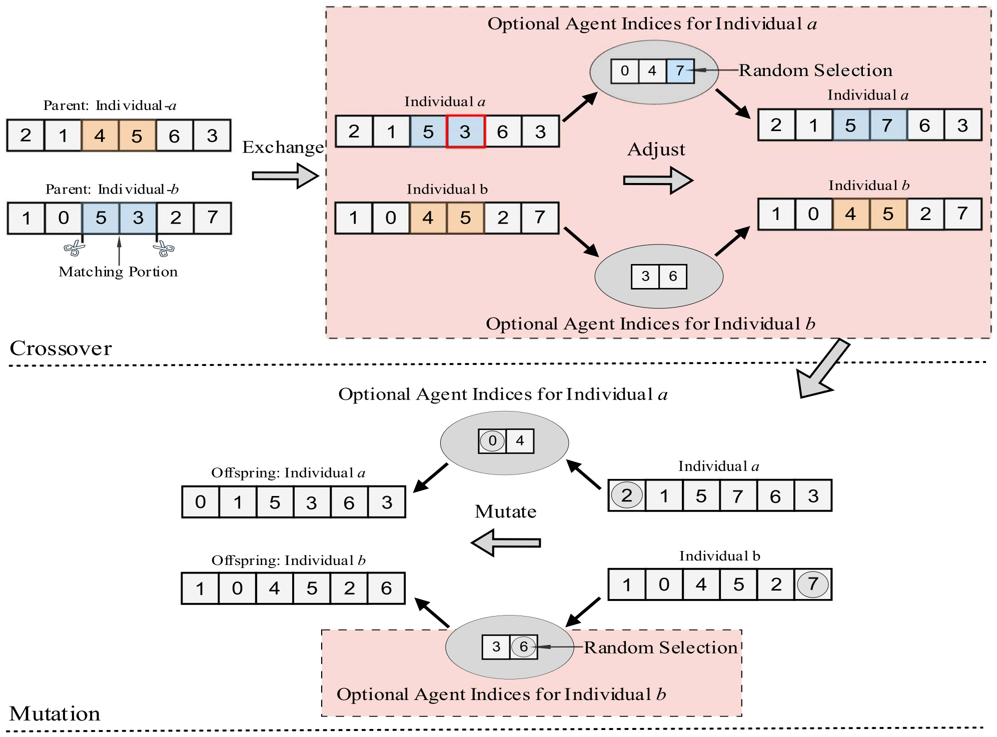


Fig. 6. Simple case to illustrate procedures of parents generating the offspring in the GRA-NSGA-II algorithm. Note: The dashed pink squares highlighting the differences between the GRA-NSGA-II algorithm and the original NSGA-II. The set of optional agent indices for an individual is $a-\psi$.

offspring populations into a new population and select individuals in the newly generated population within a front, as determined by the crowding distance.

C. Solving Pareto Solution Set in the BGRA++ Model

Drawing upon Sections IV-A and IV-B, Algorithm 3 presents the process of employing the GRA-NSGA-II algorithm to solve the BGRA++ model. More specifically, it demonstrates the procedure of calculating the Pareto solution set Φ, which is a vital input parameter for Algorithm 2. To further elucidate Algorithm 3, we introduce the following additional symbols:

1) Symbol $\tilde{\sigma}$ indicates the range for σ in (13). That is, $\tilde{\sigma} = \{\sigma_1, \sigma_2, \ldots, \sigma_K, \sigma^p\}$.
2) Symbol t represents the iteration times for the GRA-NSGA-II.

With the Pareto solution set Φ obtained by Algorithm 3, we can leverage Algorithm 2 to search for the optimal solution of the BGRA++ model. For example, once the *PF*s of Φ in Fig. 4 are determined, we can find the optimal solution, as highlighted in Fig. 4, in accordance with Algorithm 2.

Finally, we utilize Algorithm 3 to calculate the Pareto solution set Φ in the real-world scenario mentioned in Section II, i.e., the PFs in Fig. 6, and leverage Algorithm 2 to find the optimal solution for the real-world scenario, as shown in Fig. 7. For clarity of presentation, we have normalized the two objective functions, σ_1 and σ_2. In Fig. 7, the light blue triangular area frames the range of solution tolerances acceptable to the decision-maker. The five different icons represent the *PF*s obtained through the GRA-NSGA-II algorithm, corresponding

Algorithm 3: Solving the Pareto Solution Set Φ

```
Input:
➢ The Pareto set about role range vector $\Pi$
➢ Symbol $\tilde{\sigma}$
➢ Iteration times for GRA-NSGA-II $t$
Output:
➢ Pareto solution set $\Phi$
Begin
1:  $\Phi \leftarrow \emptyset$; /* Initialization */
2:  for $L$ in $\Pi$ do
3:      Initialize the parent population $P_0$ for $\tilde{\sigma}$ and current $L$
        with (13);
4:      With $P_0$ and the cross-mutation and merging methods
        mentioned in Section IV-B, the final population $P_t$ is
        obtained through $t$ rounds of iterations;
5:      Select the proper individuals in $P_t$ within a Pareto
        front based on their crowding distances, compute
        their corresponding objective values, and record
        these values into $\Phi$;
6:  end for
7:  return $\Phi$;
end
```

to the five different role range vectors L in Π. Ultimately, since all offspring generated by the GRA-NSGA-II algorithm satisfy (4) and (5), the solutions produced are represented by the outermost green cross icons, which are the Pareto points closest to the decision-maker's preference curve. The vector L

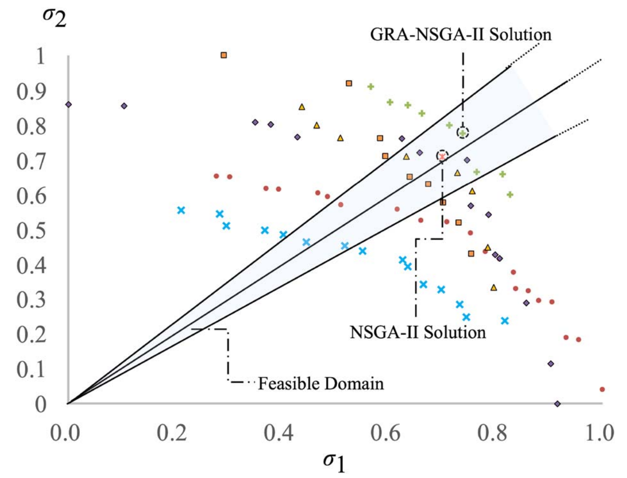


Fig. 7. Obtained PFs in the real-world scenario. Note: PFs composed of different shapes represent the results at different Ls in Π. For display purposes, here we normalize σ_1 and σ_2.

corresponding to the solution of GRA-NSGA-II is [3, 20, 1, 1]. In contrast, the NSGA-II algorithm, generating many offspring that do not meet constraints (4) and (5), achieves solutions significantly inferior to those obtained by the GRA-NSGA-II algorithm. The vector L corresponding to the solution of NSGA-II is [5, 18, 1, 1].

V. Experimental Results

In this section, we assess the proposed BGRA++ model and its associated GRA-NSGA-II algorithm through large-scale simulation experiments. To provide benchmark results, we compare the performance of the proposed method against the NSGA-II algorithm and the state-of-the-art GRA++ model, i.e., WS-GRAP.

A. Experimental Setup

To showcase the viability and efficiency of our proposed method, we conduct large-scale random simulation experiments on a MacOS-13.1 laptop equipped with Apple M1 Max and 32GB LPDDR5, utilizing Python-3.8.5 programming on the Visual Studio Code platform.

Since the scale of the experiment primarily depends on the parameters listed in Table V, we empirically refine the range or value of these parameters as follows. In line with the latest GRA-related research [27], [38], we set the range of m to be $\{50, 51, \dots, 200\}$, and the range of n to be $\{1, 2, \dots, 50\}$. As each role requires to be played by at least one agent, we empirically set n_a as 50. That is, $n_a = \sum_{j=0}^{n-1} L[j]$. Further, to reduce the impact of the dimension differences among various objective functions, we normalize the multiobjective qualification matrix Q_k ($k \in \{1, 2, 3, \dots, K\}$) using the max–min normalization method. Regarding the decision tolerance threshold α, we empirically set it as $\pi/36$, referring to the allowable range of industrial error (approximately 10%). In addition, we randomly set the vital parameters for the GRA-NSGA-II algorithms as shown in Table V.

TABLE V
Ranges of Vital Parameters

Parameter	Definition	Range or Value
m	Number of agents	$\{50, 51, \dots, 200\}$
n	Number of roles	$\{1, 2, \dots, 50\}$
n_a	Number of required agents for roles	50
Q_k	Multi-objective qualification matrix	[0, 1]
α	Decision tolerance threshold	$\pi/36$
K	Number of objective functions	$K \geq 2$, and K $\mathbb{N}^+$
$\mathcal{W}$	Decision-makers' preference vector	$w_k \in \mathcal{W}$, $\sum_{k=0}^{K-1} w_k = 1$
N	Population size	100
t	Number of iterations	100
p_c	Crossover probability	0.8
p_m	Mutation probability	0.6

B. Comparisons and Discussions

In this section, we conduct large-scale simulation experiments to verify the proposed BGRA++ model based on the range of vital parameters listed in Table V hundreds of times. First, since we utilize the mapping Π to replace the set $\mathfrak{L}$ to offer a tradeoff between the solution time and the group performance of BGRA++ models, here we examine whether this tradeoff strategy is reasonable. Considering the convenience of presentation and the common use of the NSGA-II algorithm for biobjective and triobjective optimization, here we set K to 2 and 3, and the other parameters are consistent with the range or value in Table V.

Based on the above parameter settings, Fig. 8 shows the impact of replacing $\mathfrak{L}$ with Π at the lower level of the BGRA++ model. Fig. 8(a) and 8(d), respectively, depicts how the solution time for the compared methods in biobjective and triobjective optimization scenarios varies with the increase of problem size n_a. As mentioned in Section III-B, in the lower level of the BGRA++ model, the time complexity to traverse all candidate vectors Ls in $\mathfrak{L}$ is $\binom{m-1}{n-1} = ((m-1)!)/((n-1)! \times (m-n)!)$, where m represents the number of agents and n is the number of roles. As the number of roles and agents increases, the solution time of the BGRA++ model with set $\mathfrak{L}$ can potentially expand dramatically. However, using the mapping Π to replace the set $\mathfrak{L}$ at the lower level of the BGRA++ model can significantly reduce the solution time.

Simultaneously, Fig. 8(b) and 8(e) shows the proportion of identical solutions in large-scale experiments under biobjective and triobjective optimization for the compared methods, respectively, and Fig. 8(c) and 8(f) displays the corresponding objective function values. Based on these four figures, we can observe that, under biobjective optimization, the strategy of leveraging Π to replace $\mathfrak{L}$ at the lower level of the BGRA++ model can achieve over 50% identical solutions on average. However, under triobjective optimization, even though the proportion of identical solutions for this replacement strategy is smaller, the deviation of the obtained objective function values is less than 3% [e.g., (27.69−26.91)/27.69 ≈ 0.028].

Next, we evaluate the performance of the proposed GRA-NSGA-II method against the NSGA-II algorithm and the state-of-the-art method for solving the GRA++ (i.e., the WS-GRAP algorithm [1]). For ease of presentation, we randomly generate

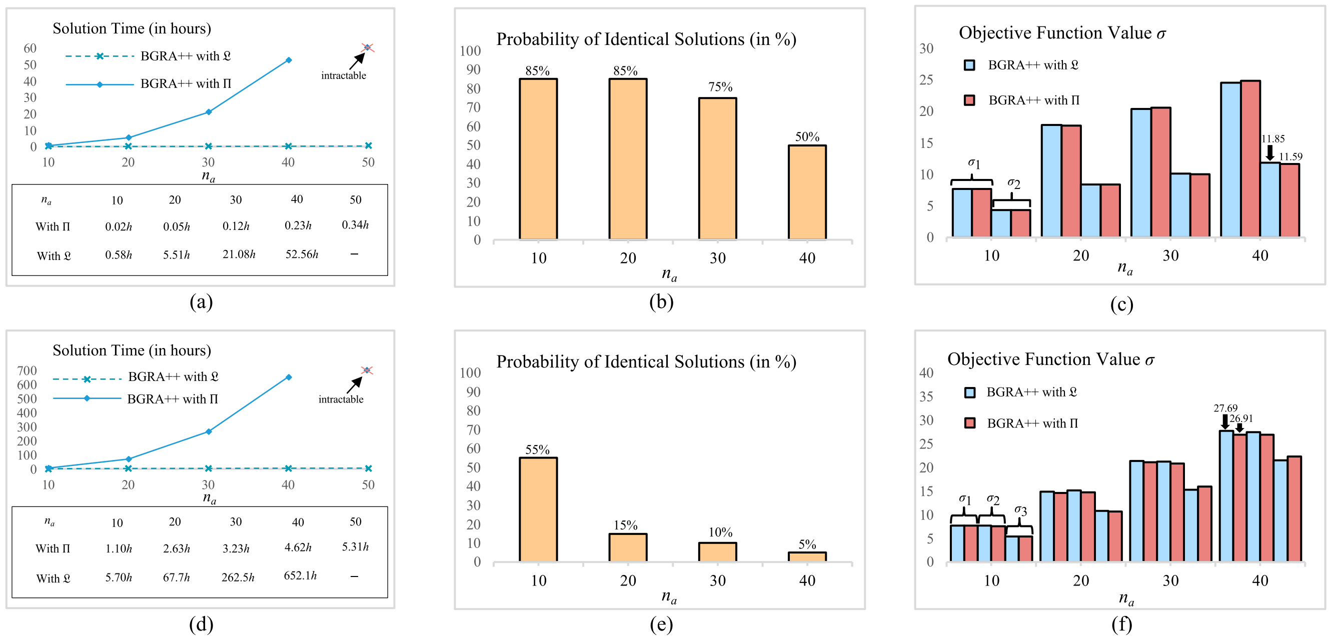


Fig. 8. Comparison results between BGRA++ with set 𝔏 and BGRA++ with mapping Π. Note: (a)–(c) Result comparison under biobjective optimization; (d)–(f) result comparison under triobjective optimization. Here, we utilize the symbol – in (a) and (d) to depict that the solution time of the BGRA++ model utilizing GRA-NSGA-II algorithm is more than seven days. (b) and (e) Illustrates the proportion of identical solutions obtained by the compared methods in large-scale experiments.

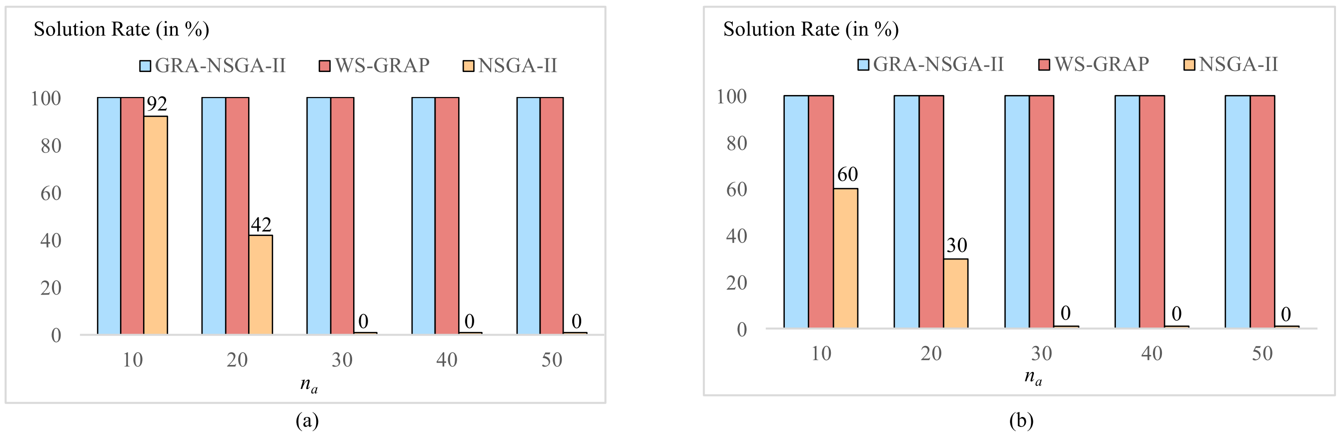


Fig. 9. Overall solution rates of GRA-NSGA-II algorithm and benchmark algorithms for different problem size n_a. Note: (a) Solution rates of different algorithms for biobjective optimization cases; (b) solution rates for triobjective optimization cases.

TABLE VI
COMPARED RESULTS WITH BASELINE METHODS UNDER TWO OBJECTIVES

Case[a]	Average Simulation Result[b]					
	WS-GRAP		NSGA-II		GRA-NSGA-II	
	σ_1[c]	σ_2	σ_1	σ_2	σ_1	σ_2
1	8.71	2.33	7.09	2.51	15.14	5.00
2	2.31	1.77	1.70	0.80	7.22	2.69
3	0.34	4.19	4.00	3.34	5.38	4.19
...	...	...	...	...	...	...
48	1.70	5.51	8.38	3.09	11.70	6.38
49	10.11	3.31	15.57	6.97	16.64	7.44
50	6.93	5.10	5.85	3.38	16.08	8.52
In Average	7.28 **(44%↓)**	3.12 **(52%↓)**	8.99 **(31%↓)**	4.32 **(34%↓)**	**12.96**	**6.50**

[a]For display purposes, only the first and last three rows are shown. [b]All values in Table IV are averages obtained from hundreds of simulation experiments. [c]σ_k represents the normalized value of the kth objective function.

50 cases for both biobjective and triobjective optimization problems, using the parameter ranges presented in Table I, and perform 100 experiments for each case, with the average value serving as the experimental result.

Fig. 9 illustrates the solution rates of the GRA-NSGA-II algorithm and the benchmark algorithms for the generated cases. It is evident that the solution rate of the NSGA-II algorithm is low, particularly when the problem size n_a becomes large. More pertinently, the NSGA-II algorithm fails to obtain effective solutions when n_a exceeds 30. This is attributed to the constraints imposed on the control variables in the BGRA++ problem. As the size of the problem, represented by n_a, increases, the NSGA-II algorithm tends to generate a significant number of offspring that violate these control variable constraints. This leads to the generation of a large number of infeasible solutions, ultimately compromising the effectiveness of the algorithm.

TABLE VII
COMPARED RESULTS WITH BASELINE METHODS UNDER THREE OBJECTIVES

Case	Average Simulation Result								
	WS-GRAP			NSGA-II			GRA-NSGA-II		
	σ_1	σ_2	σ_3	σ_1	σ_2	σ_3	σ_1	σ_2	σ_3
1	4.87	4.23	2.53	8.93	9.45	5.77	10.86	10.92	7.52
2	3.81	2.72	4.41	8.13	6.07	8.51	9.87	7.36	9.65
3	3.66	2.70	5.62	1.09	3.21	3.88	7.33	5.70	11.27
…	…	…	…	…	…	…	…	…	…
48	4.02	2.37	4.32	0.00	0.00	0.00	9.34	6.78	10.17
49	5.21	4.77	1.50	0.00	0.00	0.00	10.05	9.92	4.35
50	4.40	3.48	2.59	0.00	0.00	0.00	10.99	9.17	8.21
In Average	4.10 (57%↓)	3.62 (59%↓)	3.17 (61%↓)	3.39 (65%↓)	2.97 (66%↓)	3.40 (59%↓)	**9.59**	**8.81**	**8.22**

Consequently, the results yielded by the NSGA-II algorithm are suboptimal or even infeasible.

As for the GRA-NSGA-II algorithm, due to the control of the evolutionary process by GRA (see Section IV), it ensures that the generated offspring are valid, thereby achieving a solution rate on par with WS-GRAP. Meanwhile, Tables VI and VII present the values obtained for each objective by the compared algorithms in biobjective and triobjective optimization problems, respectively.

By examining Fig. 9(a) and Table VI, we can observe that after introducing the concept of decision tolerance, the average values for each objective of both the NSGA-II and GRA-NSGA-II algorithms are higher compared to the WS-GRAP. However, due to the large number of invalid solutions generated by the NSGA-II algorithm's offspring, the obtained results may not necessarily be better than WS-GRAP's, as in the case of case 2 in Table VI. In contrast, the GRA-NSGA-II algorithm, when applied to biobjective optimization scenarios, has improved each objective value by more than 30% compared to the benchmark algorithms. Based on Fig. 9(b) and Table VI, we can see that the solution rate of the NSGA-II algorithm is low in the triobjective optimization cases, resulting in two objectives' average values being worse than WS-GRAP's. Conversely, the GRA-NSGA-II algorithm in this scenario has improved each objective value by more than 50% on average.

From the experiment above, we can conclude the following.

1) Fig. 8 demonstrates that utilizing the mapping Π to replace the set $\mathfrak{L}$ at the lower level of the BGRA++ model can significantly reduce the solution time while marginally compromising the overall group performance.
2) Fig. 9, along with Tables VI and VII, reveal that the proposed GRA-NSGA-II algorithm at the upper level of BGRA++ can effectively solve the BGRA++ problem while existing GRA-related algorithms and evolutionary algorithms fall short.
3) By employing the proposed methods at both the lower and upper levels of the BGRA++ model, we can efficiently construct an efficient strategy for solving the GRA++ problem with iterative RN.

VI. Related Work

RN is the initial process of RBC, and is crucial for the normal execution of the entire RBC lifecycle. It is also a key component in the solution of the corresponding GRA+ and GRA++ models. There has been extensive research concerning the RN process of RBC.

Huang et al. [17] utilize the GRA+ model and their original role awareness method in the RN process of RBC to solve the last-mile assignment problem (LMAP) in crowdsourcing. Their proposed method not only assigns couriers to deliver daily orders efficiently but also improves the quality of service.

Zhu [21] delineates and formalizes the agent categorization (AC) challenge within the context of the RN process as an extended GRA+ problem. To address this, he advocates a pragmatic approach that synergizes the GRA+ algorithm with the simulated annealing (SA) heuristic. This confluence of strategies exemplifies the efficacy of RBC and GRA++ models in structuring and resolving complex collaborative and managerial quandaries.

Liu et al. [18] devise a novel role awareness method predicated on the minimum spanning tree (MST) approach for specifying signal relay points as roles during the RN stage in RBC. Based on these roles, they utilize GRA+ to model the deployment of unmanned aerial vehicles (UAVs) in disaster-stricken areas for the establishment of relay communication networks. Results from thousands of experimental simulations prove the practicability of their solutions.

However, in the research mentioned above, the RN process is static. That is, the resource requirements for each role are predetermined by the decision-makers. However in our problem, the RN process is iterative, which makes the RBC process more complex, and the methods mentioned above are not suitable for solving iterative RN in RBC.

Solving the iterative RN in RBC is essentially a bilevel multiobjective GRA (BMGRA) problem. Specifically, it is an extended bilevel multiobjective integer linear programming (BMILP) problem. To the best of our knowledge, existing methods [39], [40], [41], [42] for solving bilevel multiobjective linear programming problems are predominantly applicable to continuous-variable linear programming problems and are not directly suited for addressing the BMGRA issue. Specifically, there are two main challenges. First, due to the constraints on the control variable matrix T in GRA, existing popular multiobjective optimization algorithms [43], such as MOPSO, NSGA-II, NSGA-III, and Jaya [44], cannot be directly employed to solve this problem. Their application on BMGRA may generate a large number of offspring that violate constraints, thereby reducing the efficiency and quality of the solutions. Second, due to the nature of integer programming solutions, the optimal solution desired by decision-makers may not exist. In such a scenario, selecting a solution that aligns with the decision-makers' expectations becomes problematic.

The limitation highlights the necessity for further research to devise a new solution for the BMGRA. Therefore, the BGRA++ model with decision tolerance introduced in this article emerges as a solution to address these challenges.

VII. Conclusion

In this article, we proposed the BGRA++ model with decision tolerance to address the GRA++ problem with an iterative RN process. In the BGRA++ model, the lower level employed Pareto optimality in conjunction with GRA to compute the Pareto set concerning resource assignment for roles. At the upper level, we incorporated the GRA model with the NSGA-II algorithm to solve the BGRA++ model. Importantly, we introduced the notion of decision tolerance to facilitate the selection of an appropriate solution from the set of available options. A suite of simulation experiments were conducted to demonstrate the efficiency and robustness of the proposed methods.

The results indicated that the BGRA++ model with decision tolerance offers a versatile and efficient framework for solving complex GRA++ issues, thereby extending the applicability of GRA++ in engineering scenarios.

In future research, we intend to explore the optimal method for identifying the set Π and provide proof of its optimality. Additionally, we plan to evaluate the performance of the BGRA++ model using alternative multiobjective optimization algorithms, such as MOPSO and NSGA-III. Our goal is to identify the algorithm that offers the most effective and robust solutions, intending to establish it as the standard approach for industrial applications. Meanwhile, we hope to consider the factor of time windows and apply this model and its algorithm to the dynamic assignment problem and similar vital issues in the future. We also plan to conduct detailed comparisons with the latest models and algorithms designed to solve these problems.

Acknowledgment

The authors would like to thank the anonymous reviewers and the associate editor for their insightful comments that improved the presentation of this article.

References

[1] H. Zhu, *E-CARGO and Role-Based Collaboration: Modeling and Solving Problems in the Complex World.* Hoboken, NJ, USA: Wiley, 2022.

[2] D. Liu, Y. Yuan, H. Zhu, S. Teng, and C. Huang, "Balance preferences with performance in group role assignment," *IEEE Trans. Cybern.*, vol. 48, no. 6, pp. 1800–1813, Jun. 2018.

[3] D. Liu, B. Huang, and H. Zhu, "Solving the tree-structured task allocation problem via group multirole assignment," *IEEE Trans. Automat. Sci. Eng.*, vol. 17, no. 1, pp. 41–55, Jan. 2020.

[4] H. Zhu, D. Liu, S. Zhang, Y. Zhu, L. Teng, and S. Teng, "Solving the Many to Many assignment problem by improving the Kuhn-Munkres algorithm with backtracking," *Theor. Comput. Sci.*, vol. 618, pp. 30–41, Mar. 2016.

[5] H. Zhu, D. Liu, S. Zhang, S. Teng, and Y. Zhu, "Solving the group multirole assignment problem by improving the ILOG approach," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 47, no. 12, pp. 3418–3424, 2017.

[6] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems: Revised Reprint.* Philadelphia, PA, USA: SIAM, 2012.

[7] D. W. Pentico, "Assignment problems: A golden anniversary survey," *Eur. J. Oper. Res.*, vol. 176, no. 2, pp. 774–793, Jan. 2007.

[8] Y. Wang, B. Xin, and J. Chen, "An adaptive Memetic algorithm for the joint allocation of heterogeneous stochastic resources," *IEEE Trans. Cybern.*, vol. 52, no. 11, pp. 11526–11538, Nov. 2022.

[9] J. Turner, Q. Meng, G. Schaefer, A. Whitbrook, and A. Soltoggio, "Distributed task rescheduling with time constraints for the optimization of total task allocations in a multirobot system," *IEEE Trans. Cybern.*, vol. 48, no. 9, pp. 2583–2597, Sep. 2018.

[10] Q. Jiang, H. Zhu, Y. Qiao, Z. He, D. Liu, and B. Huang, "Agent evaluation in deployment of multi-SUAVs for communication recovery," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 52, no. 11, pp. 6968–6982, Nov. 2022.

[11] H. Ma, H. Zhu, K. Li, and W. Tang, "Collaborative optimization of service composition for data-intensive applications in a hybrid cloud," *IEEE Trans. Parallel Distrib. Syst.*, vol. 30, no. 5, pp. 1022–1035, May 2019.

[12] H. Zhu, "Computational social simulation with E-CARGO: Comparison between collectivism and individualism," *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 6, pp. 1345–1357, Dec. 2020.

[13] H. Zhu, "Avoiding conflicts by group role assignment," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 46, no. 4, pp. 535–547, Apr. 2016.

[14] H. Zhu, "Avoiding critical members in a team by redundant assignment," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 7, pp. 2729–2740, Jul. 2020.

[15] L. Liang, J. Fu, H. Zhu, and D. Liu, "Solving the team allocation problem in crowdsourcing via group multirole assignment," *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 3, pp. 843–854, Jun. 2023.

[16] F. Wen, H. Zhu, and D. Liu, "Defrost period allocation with flexible intervals via group role assignment," i*n Proc. IEEE 25th Int. Conf. Comput. Supported Cooperat. Work Des. (CSCWD)*, Hangzhou, China, 2022, pp. 414–419.

[17] B. Huang, H. Zhu, D. Liu, N. Wu, Y. Qiao, and Q. Jiang, "Solving last-mile logistics problem in spatiotemporal crowdsourcing via role awareness with adaptive clustering," *IEEE Trans. Comput. Soc. Syst.*, vol. 8, no. 3, pp. 668–681, Jun. 2021.

[18] D. Liu, Q. Jiang, H. Zhu, and B. Huang, "Distributing UAVs as wireless repeaters in disaster relief via group role assignment," *Int. J. Coop. Inf. Syst.*, vol. 29, nos. 1&2, pp. 2040002-1–22, 2020.

[19] H. Zhu, "Agent categorization with group role assignment with constraints and simulated annealing," *IEEE Trans. Comput. Soc. Syst.*, vol. 7, no. 5, pp. 1234–1245, Oct. 2020.

[20] Q. Jiang, H. Zhu, Y. Qiao, D. Liu, and B. Huang, "Refugee resettlement by extending group multirole assignment," *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 1, pp. 36–47, Feb. 2023.

[21] H. Zhu, "Group role assignment with constraints (GRA+): A new category of assignment problems," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 53, no. 3, pp. 1670–1683, Mar. 2023.

[22] H. Zhu and M. Zhou, "Role transfer problems and algorithms," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 38, no. 6, pp. 1442–1450, Nov. 2008.

[23] H. Zhu and M. Zhou, "Efficient role transfer based on Kuhn–Munkres algorithm," *IEEE Trans. Syst. Man, Cybern. - Part A Syst. Humans*, vol. 42, no. 2, pp. 491–496, Mar. 2012.

[24] B. Lazzerini and F. Pistolesi, "Multiobjective personnel assignment exploiting workers' sensitivity to risk," *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 48, no. 8, pp. 1267–1282, Aug. 2018.

[25] C. A. Coello Coello and M. S. Lechuga, "MOPSO: A proposal for multiple objective particle swarm optimization," in *Proc. Congr. Evol. Comput. (CEC)*, Honolulu, HI, USA, 2002, pp. 1051–1056.

[26] W. Mkaouer et al., "Many-objective software remodularization using NSGA-III," *ACM Trans. Softw. Eng. Methodol.*, vol. 24, no. 3, pp. 1–45, May 2015.

[27] Q. Jiang, D. Liu, H. Zhu, Y. Qiao, and B. Huang, "Equilibrium means equity? An E-CARGO perspective on the golden mean principle," *IEEE Trans. Comput. Soc.*, vol. 10, no. 4, pp. 1443–1454, Aug. 2023.

[28] R. Wachowiak-Smolíková and H. Zhu, "Data analytics and visualization of adaptive collaboration simulations," *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 1, pp. 84–93, Feb. 2023.

[29] H. Zhu, “Pareto improvement: A GRA perspective,” *IEEE Trans. Comput. Soc. Syst.*, vol. 10, no. 3, pp. 1241–1251, Jun. 2023.

[30] H. Zhu, “Maximizing group performance while minimizing budget,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 50, no. 2, pp. 633–645, Feb. 2020.

[31] H. Zhu and Y. Zhu, “Group role assignment with agents’ busyness degrees,” in *Proc. IEEE Int. Conf. Syst., Man, and Cybern. (SMC)*, Banff, AB, Canada, 2017, pp. 3201–3206.

[32] H. Zhu, “The most economical redundant assignment,” in *Proc. IEEE Int. Conf. Syst., Man, Cybern. (SMC)*, Bari, Italy, 2019, pp. 146–151.

[33] Q. Jiang, D. Liu, H. Zhu, Y. Qiao, and B. Huang, “Quasi group role assignment with role awareness in self-service spatiotemporal crowdsourcing,” *IEEE Trans. Comput. Soc. Syst.*, vol. 9, no. 5, pp. 1456–1468, Oct. 2022.

[34] Y. Sheng, H. Zhu, X. Zhou, and Y. Wang, “Effective approaches to group role assignment with a flexible formation,” in *Proc. IEEE Int’l Conf. Syst., Man Cybern. (SMC’14)*, San Diego, CA, USA, 2014, pp. 1426–1431.

[35] P. Flajolet and R. Sedgewick, *Analytic Combinatorics.* Cambridge, U.K.: Cambridge Univ. Press, 2009.

[36] “IBM ILOG CPLEX optimization package.” IBM. [Online]. Available: http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/

[37] H. Zhu, M. Zhou, and R. Alkins, “Group role assignment via a Kuhn-Munkres algorithm-based solution,” *IEEE Trans. Syst. Man, Cybern. Part A Syst. Humans*, vol. 42, no. 3, pp. 739–750, May 2012.

[38] H. Zhu, Y. Sheng, X. Zhou, and Y. Zhu, “Group role assignment with cooperation and conflict factors,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 48, no. 6, pp. 851–863, Jun. 2018.

[39] K. Deb and A. Sinha, “Solving bilevel multi-objective optimization problems using evolutionary algorithms,” in *Proc. Int. Conf. Evol. Multi-Criterion Optim. (EMO),* 2009, pp. 110–124.

[40] J.-A. Mejía-De-Dios, A. Rodríguez-Molina, and E. Mezura-Montes, “Multiobjective bilevel optimization: A survey of the state-of-the-art,” *IEEE Trans. Syst. Man, Cybern. Syst.*, vol. 53, no. 9, pp. 5478–5490, Sep. 2023.

[41] H. Li, Q. Zhang, Q. Chen, L. Zhang, and Y.-C. Jiao, “Multiobjective differential evolution algorithm based on decomposition for a type of multiobjective bilevel programming problems,” *Knowl.-Based Syst.*, vol. 107, pp. 271–288, Sep. 2016.

[42] A. Sinha, P. Malo, K. Deb, P. Korhonen, and J. Wallenius, “Solving bilevel multicriterion optimization problems with lower level decision uncertainty,” *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 199–217, Apr. 2016.

[43] S.-C. Liu, Z.-G. Chen, Z.-H. Zhan, S.-W. Jeon, S. Kwong, and J. Zhang, “Many-objective job-shop scheduling: A multiple populations for multiple objectives-based genetic algorithm approach,” *IEEE Trans. Cybern.*, vol. 53, no. 3, pp. 1460–1474, Mar. 2023.

[44] Y. Pan, K. Gao, Z. Li, and N. Wu, “Solving biobjective distributed flow-shop scheduling problems with lot-streaming using an improved Jaya algorithm,” *IEEE Trans. Cybern.*, vol. 53, no. 6, pp. 3818–3828, Jun. 2023.

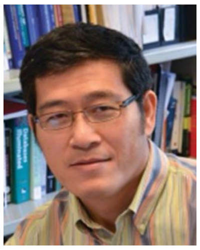
Qian Jiang is currently working toward the Ph.D. degree in intelligent science and system with Macao Institute of Systems Engineering, Macau University of Science and Technology, Macao, China.

Recently, he has published seven regular papers in IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, and *International Journal of Cooperative Information Systems*, respectively. His research interests include human–machine systems, computational social simulation, and scheduling and optimization. Mr. Jiang has served as a reviewer for IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING and IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS.

Dongning Liu (Senior Member, IEEE) received the Ph.D. degree in logic from Sun Yat-Sen University, Guangzhou, China, in 2007.

He has been a Full Professor with Guangdong University of Technology, China, since 2009. He is responsible for teaching artificial intelligent logic and discrete math at the School of Computer Science and Technology. He is engaged in education and technology transfer on collaborative computing. He is the Vice Dean of the School of Computer Science and Technology. He has published more than 50 papers on computer magazines and international conferences

Dr. Liu is a reviewer for IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, and so on. He is a Member of IEEE SMC Society, a TC Member of the Technical Committee on Distributed Intelligent Systems, a Senior Member of CCF Society, and the Vice Secretary-General and a Standing Committee Member of the Technical Committee on Cooperative Computing of China Computer Federation.



Haibin Zhu (Senior Member, IEEE) received the B.Sc. degree in computer engineering from the Institute of Engineering and Technology, China, in 1983, and the M.Sc. and Ph.D. degrees in computer science from the National University of Defense Technology (NUDT), China, in 1988 and 1997, respectively.

He is a Full Professor and a Coordinator of the Computer Science Program, the Founding Director of the Collaborative Systems Laboratory, and a Member of the University Budget Plan Committee, Arts and Science Executive Committee, Nipissing University, North Bay, ON, Canada. He is also an Affiliate Professor with Concordia University and an Adjunct Professor with Laurentian University, Canada. He was the Chair of the Department of Computer Science and Mathematics, Nipissing University (2019–2021); a Visiting Professor and a Special Lecturer with the College of Computing Sciences, New Jersey Institute of Technology, USA (1999–2002); and a Lecturer, an Associate Professor, and a Full Professor with NUDT (1988–2000). He has accomplished (published or in press) over 230 research works including 30+ IEEE Transactions articles, six books, five book chapters, four journal issues, and four conference proceedings.

Dr. Zhu is a Member-at-Large of the Board of Governors and a Co-Chair of the Technical Committee of Distributed Intelligent Systems of IEEE Systems, Man and Cybernetics (SMC) Society (SMCS), the Editor-in-Chief of *IEEE SMC Magazine*, an Associate Editor (AE) of IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS: SYSTEMS, IEEE TRANSACTIONS ON COMPUTATIONAL SOCIAL SYSTEMS, *Frontiers of Computer Science*, and *IEEE Canadian Review*. He was an Associate Vice President (AVP) of Systems Science and Engineering (SSE) (2021) and IEEE SMCS, a Program (Co-)Chair for many international conferences, and a PC Member for 130+ academic conferences. He is the Founding Researcher of Role-Based Collaboration and Adaptive Collaboration and the Creator of the E-CARGO model. His research monograph E-CARGO and Role-Based Collaboration can be found https://www.amazon.com/CARGO-Role-Based-Collaboration-Modeling-Problems/dp/1119693063.

Shijue Wu was born in May 1999. She received the B.S. degree in computer science and technology from the School of Computer Science and Technology, Guangdong University of Education, Guangzhou, China, in 2019, and the M.S. degree in software engineering from the School of Computer Science and Technology, Guangdong University of Technology, Guangzhou, China, in 2022.

Currently, she is working as a Project Management Engineer with Guangdong Science & Technology Infrastructure Center, Department of Science and Technology of Guangdong Province, Guangzhou, China. She has published one workshop paper in 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD). Her research interests include human–machine systems, resource management, and computational social simulation.

Naiqi Wu (Fellow, IEEE) received the B.S. degree in electrical engineering from Anhui University of Technology, Huainan, China, in 1982, and the M.S. and Ph.D. degrees in systems engineering from Xi'an Jiaotong University, Xi'an, China, in 1985 and 1988, respectively.

He joined Macau University of Science and Technology, Taipa, Macau, China, in 2013, where he is currently a Professor with the Institute of Systems Engineering. From 1988 to 1995, he was with Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, China. From 1995 to 1998, he was with Shantou University, Shantou, China. He moved to Guangdong University of Technology, Guangzhou, China, in 1998. He is the Author or a Co-Author of one book, five book chapters, and more than 130 peer-reviewed journal articles. His research interests include production planning and scheduling, manufacturing system modeling and control, discrete event systems, Petri net theory and applications, intelligent transportation systems, and energy systems.

Xin Luo (Senior Member, IEEE) received the B.S. degree in computer science from the University of Electronic Science and Technology of China, Chengdu, China, in 2005, and the Ph.D. degree in computer science from Beihang University, Beijing, China, in 2011.

He is currently a Professor in data science and computational intelligence with the College of Computer and Information Science, Southwest University, Chongqing, China. He has authored or coauthored over 200 papers (including over 110 IEEE Transactions papers) in the areas of his interests. His research interests include big data analysis and graph learning.

Dr. Luo was the recipient of the Outstanding Associate Editor Award from IEEE/CAA JOURNAL OF AUTOMATICA SINICA in 2020. He is currently an Associate Editor of IEEE/CAA JOURNAL OF AUTOMATICA SINICA and IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS. His page is https://scholar.google.com/citations?user=hyGlDs4AAAAJ&hl=zh-TW.

Yan Qiao (Senior Member, IEEE) received the B.S. and Ph.D. degrees in industrial engineering and mechanical engineering from Guangdong University of Technology, Guangzhou, China, in 2009 and 2015, respectively.

He is currently an Associate Professor with the Institute of Systems Engineering, Macau University of Science and Technology, Macao, China, where he was Postdoctoral Research Associate, from 2016 to 2017. From 2014 to 2015, he was a Visiting Student with the Department of Electrical and Computer Engineering, New Jersey Institute of Technology, Newark, NJ, USA. He has over 80 publications, including one book chapter and 30+ regular papers in IEEE Transactions.

Dr. Qiao was a recipient of the QSI Best Application Paper Award Finalist of 2011 IEEE International Conference on Automation Science and Engineering, the Best Student Paper Award of 2012 IEEE International Conference on Networking, Sensing and Control, the Best Conference Paper Award Finalist of 2016 IEEE International Conference on Automation Science and Engineering, the Best Student Paper Award Finalist of 2020 IEEE International Conference on Automation Science and Engineering, and the 2021 Hsue-shen Tsien Paper Award of IEEE/CAA JOURNAL OF AUTOMATICA SINICA.