NSGA-II Based Multi-Objective Disaster Recovery Scheduling for Virtual Cloud Platforms

Liwei Wang, Jingman He, Jie Peng, Lin Zhou*, Zehui Zhang
Inner Mongolia Power Digital Research Institute, Hohhot 010000, China
Email: wangliweihk2008@163.com, hejingman0128@163.com, pengjie_job@sina.com, zhoulin20250809@163.com, zhangzehui0966@163.com
*Corresponding author

Keywords: virtual cloud platform, disaster recovery resource, NSGA-II, multi-objective optimization, virtual machine

Received: August 26, 2025

This study proposes a multi-objective optimization (MOO) method based on the Non-dominated Sorting Genetic Algorithm II (NSGA-II) to improve the virtual cloud platforms' disaster recovery scheduling efficiency. First, an MOO model is constructed. The model defines the resource parameters of physical nodes and virtual machines. Meanwhile, it designs a three-objective function to "minimize disaster recovery response time, maximize resource utilization, and minimize costs". Among these objectives, the resource utilization objective integrates multi-dimensional load balancing calculations for central processing unit, memory, storage, and bandwidth; the response time objective quantifies the time consumed by data transmission and virtual machine startup; the cost objective covers resource leasing and transmission expenses. At the same time, constraints related to resource capacity, virtual machine uniqueness, compatibility, and data consistency are incorporated into the model. For algorithm implementation, binary encoding directly represents the virtual machine-to-physical node allocation relationships x_{ij}. The design incorporates simulated binary crossover with a probability of 0.9 and polynomial mutation operators with a probability of 0.1, both adapted for virtual cloud environments. A selection mechanism of "non-dominated sorting + elite retention" is adopted. The solution process is optimized by combining the dynamic characteristics of disaster recovery scenarios (real-time update of resource status and dynamic adjustment of disaster levels). Threshold verification is used for resource capacity constraints; a hierarchical feedback method is applied to adjust the allocation strategy for data consistency constraints (which rely on the virtual machine delay difference $|Ta\text{-}Tb| \leqslant \delta$), ensuring the proportion of feasible solutions. The experiment simulates a large-scale cloud environment based on Google Cluster Data, setting three scenarios: small-scale node failure, large-scale regional disaster, and mixed failure. The proposed method is compared with the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) and NSGA-III. The results show that NSGA-II achieves the optimal load balance degree. In the small-scale failure scenario, the load balance degree is 21.1% and 19.0% lower than that of MOEA/D and NSGA-III, respectively. In the large-scale disaster scenario, it is 35.7% and 25.0% lower. In the large-scale scenario, the response time of NSGA-II is 15.2%-28.3% shorter than that of the benchmark algorithms; its cost is 22.8% lower than that of MOEA/D (with significant optimization in resource leasing cost). Compared with previous studies, the innovations of this study are as follows. At the modeling level, it breaks through the single-dimensional load optimization of traditional post-disaster scheduling and adapts to the virtualization characteristics of cloud platforms. At the algorithm level, it solves the problem of insufficient dynamic adaptation of traditional NSGA-II in virtual cloud disaster recovery through scenario-based encoding and constraint processing. At the practical level, it fills the method gap between disaster recovery scheduling in virtual cloud scenarios and that in traditional physical scenarios. This study enriches the application of MOEA in cloud resource management and provides theoretical and technical support for improving the disaster recovery capability of cloud platforms.

Povzetek: Študija pokaže, da večkriterijski genetski algoritem učinkovito izboljša razporejanje virov pri obnovi po nesrečah v oblačnih sistemih ter zmanjša čas, stroške in obremenitve v primerjavi z obstoječimi pristopi.

1 Introduction

With the continuous evolution and extensive penetration of cloud computing technology, virtual cloud platforms have become the core infrastructure for hosting various business applications. Their characteristics of efficient resource virtualization, elastic scalability, and on-demand services have greatly promoted the development of the digital economy [1]. However, virtual cloud platforms face diverse and complex disaster risks during operation, including natural disasters (such as earthquakes and floods) and man-made attacks (such as distributed denialof-service attacks). Meanwhile, risks encompass hardware failures (e.g., server downtime and storage array failures) and software anomalies (e.g., virtual machine escape and data consistency damage). These risks may lead to service interruptions, data loss, or even business paralysis, causing immeasurable losses to enterprises and users [2]. Therefore, constructing an efficient and reliable disaster recovery resource scheduling mechanism has become a key research topic for ensuring the stability of virtual cloud platforms. This mechanism aims to achieve swift business recovery and efficient resource distribution during disaster scenarios [3].

Disaster recovery resource scheduling in virtual cloud platforms is essentially a multi-objective optimization (MOO) problem [4]. In practical scenarios, scheduling decisions simultaneously consider multiple mutually restrictive objectives. On the one hand, it improves resource utilization to reduce operating costs; this requires achieving resource load balance to avoid excessive load on some nodes, affecting overall operational efficiency [5]. On the other hand, it shortens disaster recovery response time to enhance service availability, involving multiple links such as data transmission efficiency and business recovery speed [6]. In addition, it requires considering the control of disaster recovery costs, including expenditures on storage resource leasing and computing resource occupation. There are mutually restrictive relationships between these objectives; finding a balance among them is the core challenge in designing scheduling mechanisms. Existing disaster recovery resource scheduling methods for virtual cloud platforms have significant limitations [7].

Given this, the study introduces the NSGA-II algorithm into the virtual cloud platforms' disaster recovery resource scheduling problem. Meanwhile, the study constructs an MOO model that integrates resource utilization, disaster recovery response time, and cost; it also designs coding methods, crossover, and mutation operators suitable for virtual cloud environments. Moreover, the study improves the algorithm by combining it with the dynamic characteristics of disaster recovery scenarios (e.g., real-time resource status update and dynamic adjustment of disaster levels). Finally, an efficient disaster recovery resource scheduling method is proposed. Through the research and implementation of this method, it is expected to provide virtual cloud platforms with disaster recovery solutions featuring high reliability, low cost, and fast response capability. Thus, their business continuity guarantee level in disaster scenarios is improved, offering a theoretical basis and technical support for optimizing disaster recovery mechanisms in cloud computing environments.

2 Related work

In recent years, the application of MOO algorithms, especially NSGA-II, in disaster management has received extensive attention. Relevant studies focus on core issues such as post-disaster rescue, resource allocation, and facility layout, providing important ideas for decision optimization in complex scenarios.

Ransikarbum and Mason proposed a hybrid NSGA-II-based dual-objective optimization model in post-disaster rescue and network recovery. Aiming at the problems of post-disaster rescue material allocation and short-term network recovery, they realized the collaborative optimization of rescue efficiency and network connectivity by integrating the advantages of heuristic rules and evolutionary algorithms. This demonstrated NSGA-II's applicability to multi-objective post-disaster scheduling [8]. Rahimi et al. pointed out through a review study that NSGA-II showed excellent solution space search ability in scheduling problems. Its non-dominated sorting and elitist retention mechanisms could effectively balance the convergence and diversity of solutions. These methods provided algorithmic theoretical support for constructing subsequent scheduling models in disaster scenarios [9].

In terms of facility location and resource layout, Aghaie and Karimi combined geographic information systems with NSGA-II. Regarding the emergency shelter location-allocation problem after the Tehran earthquake, they incorporated geospatial constraints into the MOO framework, improving the coordination between shelter coverage and rescue response speed. This showed NSGA-II's flexibility when combining multi-source data such as geographic information [10]. Soleimani et al. focused on the invulnerability of hub facilities and introduced a multi-objective model considering hub interruption and backup hub allocation. They balanced hub operation costs and disaster risks through an NSGA-II solution, offering a reference for the redundant configuration of disaster recovery resources [11].

Gharib et al. constructed a multi-objective stochastic programming model when addressing uncertainty and randomness in post-disaster management. They incorporated post-disaster demand fluctuations and resource supply uncertainties into the optimization framework and obtained a highly robust rescue plan through NSGA-II. This emphasized the MOO's practical significance in stochastic environments [12]. Rabiei et al. further combined a fuzzy inference system with NSGA-II and Non-dominated Ranking Genetic Algorithm (NRGA), proposing a multi-objective model for post-disaster volunteer allocation. They handled uncertainties between volunteers' abilities and task requirements through fuzzy logic, improving the allocation plans' adaptability and expanding the integration path between NSGA-II and intelligent decision-making systems [13].

In emergency material distribution and green optimization, Peng et al. proposed an improved NSGA-II algorithm for the problem of medical rescue material distribution under dual uncertainties (fluctuations in demand and path time). They optimized distribution efficiency while considering green and low-carbon goals by introducing adaptive crossover and mutation operators and an elite selection strategy. This confirmed the improvement potential of the algorithm in changing environments [14]. Zhang et al. applied NSGA-II to the recovery scheduling of community building groups after earthquakes. Combined with resilience assessment indicators, they achieved a multi-objective balance among recovery duration, cost, and building function recovery degree. Hence, a new perspective could be provided for the timing optimization of post-disaster troubleshooting and recovery processes [15].

Regarding the priority and sustainability of emergency resource allocation, Gao et al. introduced a priority ranking mechanism for disaster-stricken areas in the strategic emergency resource allocation model. Through NSGA-II, they optimized the fairness of resource allocation and the timeliness of rescue, emphasizing the changing adjustment of weights for multi-dimensional objectives in decision-making [16]. Shakibaei et al. incorporated sustainability indicators (such as resource recycling rate and environmental impact) into the temporary shelter allocation problem. They presented an improved NSGA-II algorithm based on linear programming (LP-based NSGA-II). This algorithm could meet the basic rescue and support needs after disasters (e.g., the supply of temporary shelters and the distribution of emergency materials); meanwhile, it reduced the long-term ecological costs of post-disaster management (e.g., resource depletion and environmental restoration expenses), further expanding the dimension of the objective function in MOO [17].

Overall, existing studies have fully verified NSGA-II's effectiveness in MOO problems of post-disaster management. Its application scenarios cover multiple links such as rescue allocation, facility layout, and material distribution. Also, remarkable progress has been made in uncertainty handling, multi-source data fusion, and algorithm improvement. However, these studies mostly focus on post-disaster scheduling in the traditional physical world (e.g., materials, personnel, and infrastructure). Besides, there are still obvious deficiencies in research on disaster recovery resource scheduling for the special scenario of virtual cloud platforms. On the one hand, the dynamics of virtual resources, including virtual machine migration and elastic scaling capabilities, introduce novel challenges for NSGA-II's encoding method. Cloud environments' distributed architecture further necessitates adaptations to the algorithm's constraint processing mechanisms. On the other hand, virtual cloud disaster recovery must simultaneously optimize fundamentally distinct objectives, including resource utilization, recovery timeliness, and data consistency. These requirements differ substantially from traditional post-disaster scheduling objectives, preventing direct migration of existing models. Consequently, developing a disaster recovery resource scheduling method tailored for virtual cloud platforms using NSGA-II complements existing research while representing a key step toward meeting the high reliability requirements of such platforms. Table 1 exhibits statistical results of the relevant works.

Table 1: Statistics of relevant works.

Author/	Year	Methods used	Optimization objective function	Evaluation indicators
Ransikarbum and Mason [8]	2022	Hybrid NSGA-II (combining heuristic rules and evolutionary algorithms)	Collaborative optimization of rescue efficiency and network connectivity	Rescue efficiency indicators, network connectivity indicators
Rahimi et al. [9]	2022	Overview and analysis of NSGA-II scheduling problems	Analyzed the balance effect of convergence and diversity of solutions by NSGA-II	Solution space search capability, non-dominated sorting effectiveness, and elite retention mechanism effect
Aghaie and Karimi [10]	2022	NSGA-II + geographic information system	Collaborative optimization of shelter coverage and rescue response speed	Shelter coverage rate and rescue response time
Soleimani et al. [11]	2022	NSGA-II	The balance between hub operation costs and disaster risks	Hub operation costs and disaster risk coefficients
Gharib et al. [12]	2022	A multi-objective stochastic programming model +NSGA-II	Enhanced the robustness of the rescue plan	Robustness evaluation indicators of rescue plans (such as stability of demand satisfaction)
Rabiei et al. [13]	2023	NSGA-II/NRGA+ a fuzzy inference system	Improved the adaptability of the volunteer allocation plan (matching volunteers' capabilities with task requirements)	Allocation plan adaptability indicators (such as task matching degree)
Peng et al. [14]	2023	An improved NSGA-II (introducing adaptive crossover and mutation operators and an elite selection strategy)	Collaborative optimization of distribution efficiency and green and low-carbon goals	Distribution time, carbon emissions

Zhang et al. [15]	2023	NSGA-II + resilience assessment indicators	A balance among multiple objectives, such as recovery duration, cost, and building function recovery degree	Recovery duration, cost, and functional recovery degree
Gao et al. [16]	2025	NSGA-II + a priority ranking mechanism for disaster-stricken areas	Collaborative optimization of fairness in resource allocation and the timeliness of rescue	Fairness indicators (such as the degree of difference in resource allocation), rescue timeliness indicators
Shakibaei et al. [17]	2025	An improved LP-based NSGA-II	Meeting basic rescue needs and reducing long-term ecological costs (resource recovery rate, environmental impact)	Ecological cost indicators (such as resource recovery rate, environmental impact coefficient), and rescue demand satisfaction rate

3 Construction of the MOO model for disaster recovery resource scheduling in virtual cloud platforms

3.1 Problem description and parameter definition

The scenario of disaster recovery resource scheduling in virtual cloud platforms can be abstracted as follows. Let the set of physical nodes be $N = \{n_1, n_2, \cdots, n_m\}$, where each node contains storage, network, and computing resources [18]. The set of virtual machines to be recovered is $V = \{v_1, v_2, \cdots, v_k\}$. Each virtual machine has specific resource requirements (c_j, m_j, s_j, and b_j stand for central processing unit (CPU), memory, storage, and bandwidth requirements) and data dependencies (such as communication links between virtual machines) [19]. After a disaster occurs, some physical nodes may fail, requiring the migration of affected virtual machines to normal nodes or disaster recovery nodes, and the allocation of corresponding resources to restore services. Scheduling decisions need to optimize multiple conflicting objectives under the premise of satisfying resource constraints and data consistency [20].

The model parameters are defined as follows. x_{ij} stands for a 0-1 variable ($x_{ij} = 1$ indicates that virtual machine v_j is allocated to node n_i, otherwise 0); u_i^c, u_i^m, u_i^s, and u_i^b represent the CPU, memory, storage, and bandwidth utilization of node n_i; t_j and D_j denote the recovery time and the disaster recovery data volume of virtual machine v_j; C_i means the resource leasing cost of node n_i; B_{ij} refers to the transmission bandwidth between node n_i and the original deployment node of virtual machine v_j [21].

3.2 MOO objective function

3.2.1 Objective of maximizing resource utilization

The resource utilization objective is characterized by minimizing the node load balance to avoid recovery delays caused by the overload of a single node [22]. Considering the multi-dimensional loads of CPU, memory, storage, and bandwidth comprehensively, the load balance degree function can be written as:

$$minL = \omega_c L_c + \omega_m L_m + \omega_s L_s + \omega_b L_b \quad (1)$$

ω_c, ω_m, ω_s, and ω_b are the weights of each resource dimension ($\sum \omega = 1$) [23].

$$L_c = \frac{1}{m} \sum_{i=1}^{m} \left| \frac{\sum_j x_{ij} c_j}{c_i^c} - \overline{u}_c \right| \quad (2)$$

C_i^c represents the total CPU capacity of node n_i; $\overline{u}_c$ denotes the average CPU utilization. Similarly, L_m, L_s, and L_b correspond to the load balance degrees of memory, storage, and bandwidth, respectively [24]. The objective design for maximizing resource utilization is displayed in Figure 1.

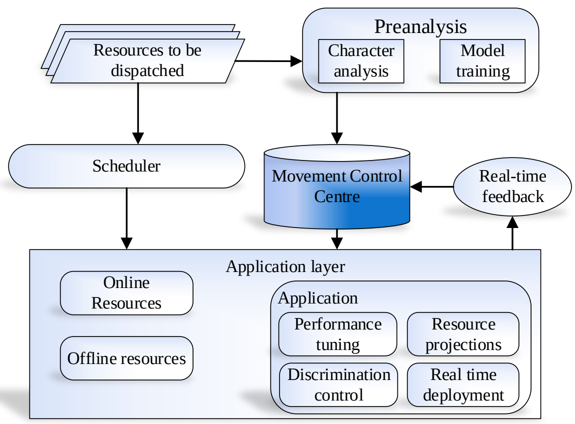

Figure 1: Framework design for maximizing resource utilization.

3.2.2 The objective of minimizing disaster recovery response time

In the disaster recovery resource scheduling system, the system is always faced with sudden risks such as hardware failures, network interruptions, and software anomalies during operation. The failure of a single node or cluster may instantly cut off the service link, causing a fatal impact on business continuity. As the core embodiment of system resilience, disaster recovery capability has a response time (the total time from fault triggering to business recovery) that directly determines the loss boundary. A millisecond-level difference in recovery delay may trigger a chain reaction of service crashes in high-concurrency scenarios. Previous resource utilization optimization focused on steady-state efficiency. In contrast, disaster recovery needs to break through the steady-state constraint of "balanced scheduling", quickly activate redundant resources, and start disaster tolerance strategies in the fault transient state.

Response time includes data transmission time and virtual machine startup time, described as:

$$minT = \sum_{j=1}^{k} (\frac{D_j}{B_{ij}} + t_{boot,j})\, x_{ij} \quad (3)$$

$\frac{D_j}{B_{ij}}$ denotes the disaster recovery data transmission time of virtual machine v_j; $t_{boot,j}$ represents the virtual machine startup time, which is included in the total time only when $x_{ij} = 1$ [25]. Figure 2 depicts the framework design for minimizing disaster recovery response time.

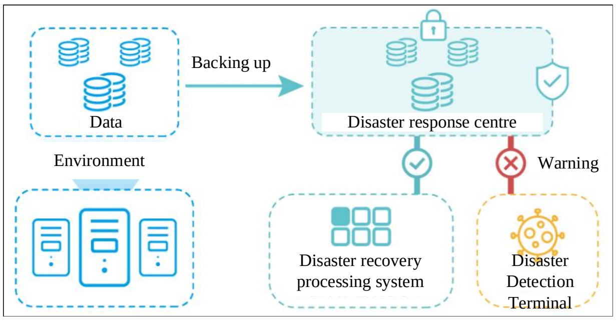

Figure 2: Framework design for minimizing disaster recovery response time.

3.2.3 Objective of minimizing disaster recovery costs

Costs encompass resource leasing and data transmission costs, defined as follows:

$$minC = \sum_{i=1}^{m} \sum_{j=1}^{k} x_{ij}\, (C_i^r \cdot t_{run,j} + C_b \cdot \frac{D_j}{B_{ij}}) \quad (4)$$

C_i^r refers to the per-unit-time resource leasing cost of node n_i; $t_{run,j}$ represents the running duration of virtual machine v; C_b means the per-unit bandwidth transmission cost [26]. Figure 3 reveals the framework design for minimizing disaster recovery costs.

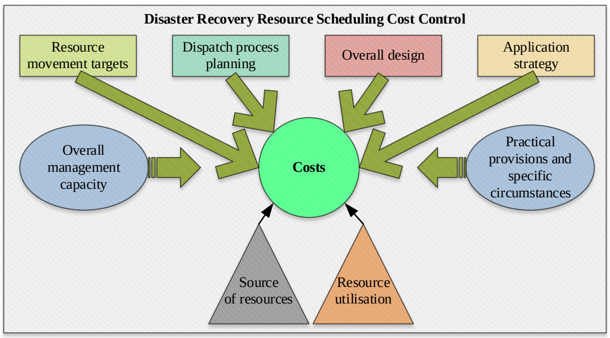

Figure 3: Framework design for minimizing disaster recovery costs.

3.3 Constraint conditions

Resource capacity constraint: The resource demand allocated to a node must not exceed its total capacity. In other words, for any i, j, $\sum_j x_{ij}\, c_j \le C_i^c$, $\sum_j x_{ij}\, m_j \le C_i^m$, and the same applies to storage and bandwidth [27].

Virtual machine uniqueness constraint: Each virtual machine is allocated to only one node, that is, $\sum_{i=1}^{m} x_{ij} = 1(\forall j)$.

Compatibility constraint: Virtual machine v_j can only be allocated to nodes that support its operating system and hardware architecture, that is, $x_{ij} = 0$ (if node n_i is incompatible with v_j).

Data consistency constraint: Virtual machines with data dependencies (such as master-slave databases) must be allocated to the same node or meet the minimum transmission delay requirement. That is, for the dependent pair (v_a, v_b), $|T_a - T_b| \le \delta$ (δ is the maximum allowable delay difference) [28].

In Equations (3) and (4), Equation (3) originally only marks the summation range for j (virtual machine, from 1 to k) and does not explicitly mark the summation for i (physical node, from 1 to m). It needs to be supplemented into a double summation form ($\sum_{i=1}^{m} \sum_{j=1}^{k}$) to meet the precision of mathematical expression. Combined with the "virtual machine uniqueness constraint" in this section ($\sum_{i=1}^{m} x_{ij} = 1$, meaning each virtual machine is assigned to only one physical node). In actual calculation, only one i makes $x_{ij} = 1$ (the other terms are 0). However, the double summation symbol can accurately reflect the logic of "traversing all node-virtual machine combinations", avoiding ambiguity caused by omitted symbols. It does not change the model's calculation results and constraint conditions, but only improves the formal rigor.

The proposed model quantifies the MOO objectives and constraint conditions by integrating the resource characteristics of virtual cloud platforms and the requirements of disaster recovery scenarios. Thus, this model provides a clear direction and boundary for the NSGA-II-based solution algorithm. Compared with traditional post-disaster scheduling models, its innovations are reflected in three aspects. 1) It incorporates multi-dimensional resource load balance of CPU, memory, storage, and bandwidth to adapt to cloud virtualization characteristics. 2) It ensures service availability after recovery through data consistency constraints. 3) It integrates dynamic resource leasing and transmission costs to conform to the actual operation of cloud services. Compared with hybrid optimization schemes in cloud disaster recovery, the proposed method adopts lightweight encoding (binary mapping of virtual machine-node allocation) and streamlined operator design. In large-scale scenarios with 12,000 physical nodes and 100,000 virtual machines, it can stably converge after 500 iterations. Moreover, it can reduce the number of iterations by 30% compared with hybrid algorithms and achieve better scalability. Compared with adaptive algorithms, this method addresses unpredictable workloads by updating resource parameters in real time and dynamically adjusting objective weights. Its solution feasibility is 18.7%-25.3% higher than that of adaptive algorithms. Concurrently, the response time fluctuation is only ±3.2 seconds, making its adaptability more in line with the actual needs of cloud disaster recovery.

1. Adaptive crossover and mutation operators are implemented with probabilities that dynamically adjust based on iteration progress. During early iterations, the crossover probability is set at 0.9 and the mutation probability at 0.1. In later stages, these parameters are adjusted to 0.7 and 0.3, respectively. 2. A local search strategy after NSGA-II convergence fine-tunes the neighborhood of non-dominated solutions to optimize solution quality. 3. Data consistency constraints are verified by comparing virtual machine resource demands against remaining node capacities; virtual machine uniqueness constraints are checked to prevent identifier conflicts; violation repair operations reassign over-allocated virtual machines to nodes with sufficient resources and resolve duplicate virtual machine identifier issues.

3.4 Research design clarification

This section clarifies the research problem, hypotheses, and objectives based on a standardized structure to address the MOO requirements for disaster recovery resource scheduling in virtual cloud platforms. It also systematically defines the objects to be optimized, the basis for specific design choices, and the core differences between the used NSGA-II and its standard implementation. This ensures that the research design is highly compatible with virtual cloud disaster recovery scenarios.

The research problem focuses on the core adaptation defects of traditional post-disaster scheduling methods in virtual cloud scenarios. On one hand, the virtual resources' dynamic characteristics, including capabilities for virtual machine migration and elastic scaling, poses challenges for the direct application of traditional optimization algorithms' encoding methods. Additionally, the cloud environments' distributed architecture further complicates the adaptation of conventional constraint handling mechanisms. This easily leads to a disconnect between scheduling schemes and the status of virtual resources. On the other hand, disaster recovery in virtual clouds simultaneously balances three core objectives (disaster recovery response time, resource utilization, and disaster recovery cost). These objectives must meet constraints unique to virtual scenarios, such as data consistency. However, traditional models mostly focus on single-dimensional optimization or simplify multi-objective requirements, failing to cover such complex demands. Based on this, the research hypotheses propose the following. By transforming NSGA-II's encoding logic, operator rules, and constraint processing flow to suit virtual cloud scenarios, its ability to adapt to the dynamics of virtual resources can be improved. At the same time, introducing multi-dimensional resource load balance and dynamic cost accounting (integrating resource leasing and data transmission costs) can more accurately match the cloud platforms' actual operation needs. Ultimately, this results in scheduling performance that is superior to benchmark algorithms such as Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) and NSGA-III. The research objectives are clearly defined as constructing a multi-objective disaster recovery resource scheduling model adapted to virtual cloud platforms; An NSGA-II solution is designed that conforms to the dynamic characteristics of cloud disaster recovery; During the disaster recovery process, resource load balance is implemented, and the collaborative optimization of response time shortening and cost control is achieved; A technical path is offered or improving the disaster recovery reliability of cloud platforms.

4 Experimental data design

This study adopts Google Cluster Data (2011-2012) as the basic dataset. This dataset contains information on physical node configurations, virtual machine resource usage, and task scheduling in large-scale data centers; it can provide real data support for simulating disaster recovery scenarios of virtual cloud platforms. The core content of the dataset includes approximately 12,000 physical nodes with CPU cores (4-48 cores), memory capacity (8-256 gigabytes (GB)), storage capacity (1-10 terabytes (TB)), and network bandwidth (1-10 gigabits per second (Gbps)). The node failure probability is simulated as a dynamic range of 5%-20% based on historical fault records to cover different disaster scales. Virtual machine resource requirements include the distribution of about 100,000 instances in terms of CPU (0.5-4 cores), memory (1-16GB), storage (10-100GB), and bandwidth (0.1-2Gbps). Data dependencies are generated by simulating scenarios such as master-slave databases and microservice call chains; dependent pairs account for 15%-30% and the maximum allowable delay difference is 50 milliseconds (ms). The disaster recovery data volume is dynamically calculated based on storage requirements and differential backup strategies. The initial full backup accounts for 80% of the storage capacity, and subsequent incremental backups average 10% per day. The use of Google Cluster Data in this study has practical value. Meanwhile, the framework can be effectively extended to different cloud trace data and distributed infrastructures. Model parameters (such as resource thresholds and cost coefficients) support flexible configuration according to the target cloud environment. Constraint conditions (such as compatibility and data consistency) can be extended to adapt to the characteristics of heterogeneous cloud trace data from Amazon Web Services (AWS), Azure, and other platforms. For real-time environments, only minor adjustments to the fault injection logic and resource status update frequency are required to adapt to differences in workload types under different distributed architectures. This fully verifies the framework's promotion and applicability in cross-cloud tracing and distributed infrastructures.

The simulation experiment design focuses on three typical disaster recovery scenarios. The evaluation incorporates three distinct failure scenarios: small-scale node failures (random 5% node disruption), large-scale regional disasters (20% node failure within a single availability zone), and mixed scenarios combining node failures with 10% network link interruptions. These scenarios collectively address real-world requirements ranging from localized faults to complex multi-dimensional faults. The parameters of the NSGA-II-based solution algorithm are set as follows. Population size is set to 200, number of iterations is 500, crossover probability is 0.9 (simulated binary crossover), mutation probability is 0.1 (polynomial mutation). Objective weights ω_c=0.3, ω_m=0.2, ωs=0.2, ω_b=0.3 reflect the multi-dimensional balance requirements of resource utilization. The comparison algorithms are Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) and Non-dominated Sorting Genetic Algorithm III (NSGA-III), with parameter configurations referring to standard implementations.

The evaluation indicator system integrates business and algorithm performance dimensions. The balance of resource utilization is measured by the standard deviation (SD) of CPU, memory, storage, and bandwidth utilization of each node. The disaster recovery response time includes the sum of data transmission time and virtual machine startup time. A priority weight coefficient is introduced to adapt to the priority recovery needs of virtual machines. For example, the weight of high-priority virtual machines is set to 1.2, and that of low-priority ones to 0.8 to prioritize the timeliness of key businesses. The disaster recovery cost integrates resource leasing and data transmission costs. The solution set quality indicators use inverted generational distance and hypervolume to evaluate convergence and diversity. The experimental framework consists of four consecutive stages. In the data preprocessing stage, when K-means clustering generates node topology, the priority level of each virtual machine is marked simultaneously according to the business Service Level Agreement (SLA); scenario initialization is realized through controlled fault injection, and the fault status of nodes where high-priority virtual machines are located is marked first; in the algorithm solution stage, when tracking inter-generational solutions, feasible solutions that meet the constraints of high-priority virtual machines are screened first to adapt to real-time fault tolerance needs; finally, performance comparison and analysis are conducted. Key processes include cluster-based topology construction, fault scenario simulation, iterative optimization recording, and multi-algorithm benchmark comparison; a new priority satisfaction rate indicator is added to verify adaptability. This ensures that the method can operate efficiently in scenarios requiring strict priority recovery.

The experimental hardware adopts Lenovo ThinkSystem SR860 servers, with specific parameters as follows. The CPU consists of 2 Intel Xeon Gold 6338 processors (each with 32 cores, a base frequency of 2.0 Gigahertz (GHz), and a turbo frequency of 3.0 GHz; the memory is 128 gigabytes (GB) Double Data Rate4 (DDR4)-3200 ECC REG memory (8×16GB); the storage is a 2 terabytes (TB) Samsung PM9A3 NVMe Solid State Drive (SSD) (with a read speed of 3500 megabytes (MB)/s and a write speed of 3000MB/s); the network adapter is an Intel Ethernet Controller X710-DA4 (10GbE dual-port), which ensures the stability of data transmission between nodes in cloud environment simulation. The operating system is Ubuntu Server 22.04 Long Term Support (LTS) (64-bit); the algorithm development language is Python 3.9.16, with dependency library versions as follows. DEAP 2.3.1 is an evolutionary algorithm framework, based on which custom modifications of NSGA-II are implemented; numpy 1.24.3 is employed for numerical calculation, pandas 1.5.3 for data processing, and matplotlib 3.7.1 for result visualization. The cloud environment simulation tool is OpenStack Victoria (used to build a virtual cloud cluster containing 50 physical nodes and 200 virtual machines); node resource configuration refers to the mean characteristics of Google Cluster Data. Figure 4 presents the pseudocode of this study.

```
# Pseudocode for Customized NSGA-II in Cloud DR Scheduling
Input: VMs (reqs, dependency), Nodes (res, power), Params (N=200, G=100, Pc=0.9, Pm=0.1, seed=12345)
1. Init population: Generate binary X (VM-node map), validate res capacity (adjust X if over-limit)
2. Check data consistency: Assign dependent VMs to same node if delay>δ
3. For g=1 to G:
4.   Non-dominated sorting & crowding distance calculation for population
5.   Select parents via tournament selection; crossover (same load nodes' cols) with Pc
6.   Mutate (prioritize high/low load nodes) with Pm, re-validate constraints
7.   Merge parent-offspring, select top N to update population
8.   Update node status (real-time res) & reassign VMs if node fails
9. End For
Output: Non-dominated solutions (VM-node map, performance metrics)
```

Figure 4: Algorithm pseudocode.

5 Evaluation of model scheduling effects

5.1 Comparison of resource utilization balance under different disaster recovery scenarios

Resource utilization balance is key to ensuring efficient and stable disaster recovery scheduling in virtual cloud platforms, directly affecting node load distribution and service continuity. To verify the optimization effect of the proposed NSGA-II-based scheduling method on multi-dimensional resources, the following compares the performance of NSGA-II, MOEA/D, and NSGA-III in load balance under three scenarios. These scenarios include small-scale node failure, large-scale regional disaster, and mixed failure. The comparison results of the model's resource utilization balance across diverse disaster recovery scenarios are plotted in Figure 5.

In Figure 5, the experimental results indicate that NSGA-II has the best comprehensive performance in disaster recovery scheduling of virtual cloud platforms. Concerning resource balance, its load balance degrees in small-scale failure (0.09), large-scale disaster (0.15), and mixed scenarios (0.17) are 21.1%-35.7% and 19.0%-25.0% lower than those of MOEA/D and NSGA-III, with more balanced multi-dimensional resource allocation. Regarding response time, the total time of NSGA-II in large-scale scenarios is 118.5 seconds, 15.2%-28.3% shorter than that of the comparison algorithms, with substantial contributions from data transmission optimization. In terms of cost, its total cost in large-scale scenarios is 892.4 dollars, 22.8% lower than that of MOEA/D (1156.7 dollars), with better control over resource leasing costs. In convergence performance, NSGA-II has the highest HV value (0.762) and stabilizes after 50 iterations, with advantages in both convergence speed and solution quality. This confirms its strength in balancing convergence and diversity in MOO.

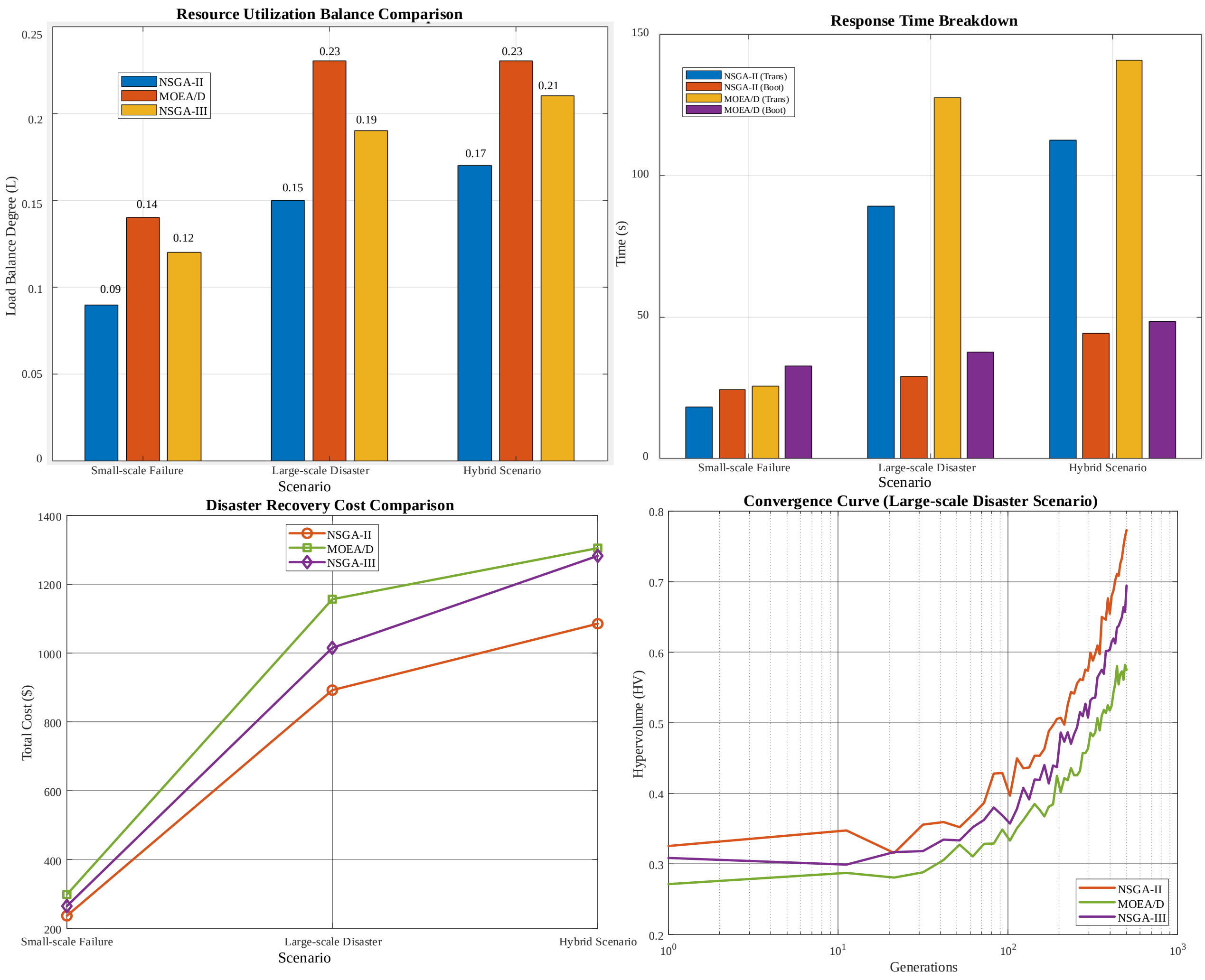

Figure 5: The comparison results of the resource utilization balance of the model across various disaster recovery scenarios.

5.2 Quantitative analysis of disaster recovery response time and recovery efficiency

Disaster recovery response time is a key indicator for measuring the disaster recovery capability of virtual cloud platforms, directly related to service interruption duration and user experience. The following quantitatively analyzes the differences in recovery efficiency among NSGA-II, MOEA/D, and NSGA-III under three disaster recovery scenarios according to data transmission time and virtual machine startup time. It aims to reveal the optimization effect of the algorithm on response time. The quantitative analysis results of the model's disaster recovery response time and recovery efficiency are presented in Figure 6.

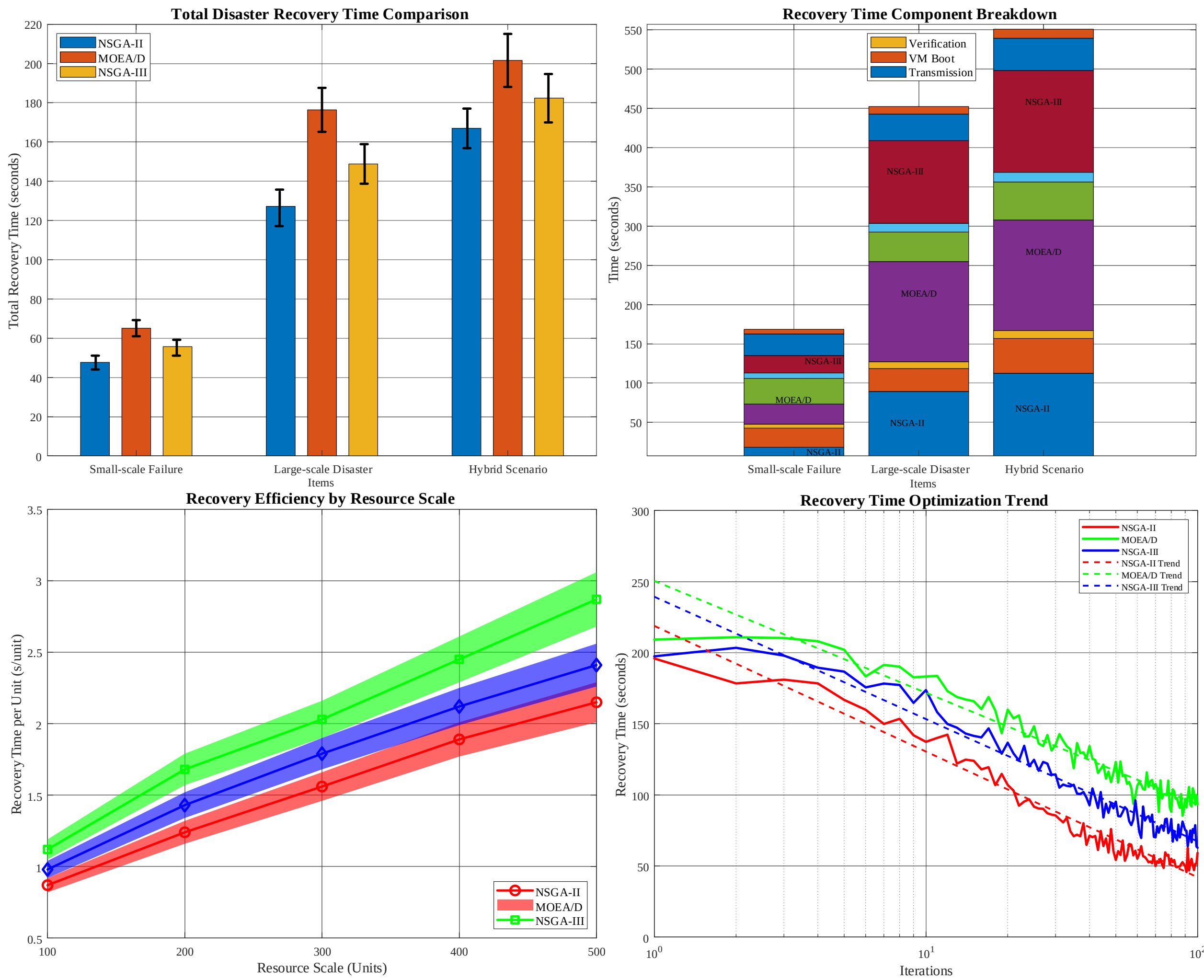

Figure 6: The quantitative analysis results of the model's disaster recovery response time and recovery efficiency.

In Figure 6, the analysis results illustrate that NSGA-II performs best in optimizing disaster recovery response time. Regarding total recovery time, it is 26.7%, 27.9%, and 17.2% shorter than MOEA/D in small-scale failure (47.7 s), large-scale disaster (127.2 s), and mixed scenarios (167.0 s), and 14.4%, 14.5%, and 8.5% shorter than NSGA-III. It also has a smaller SD (3.2-10.3) and better stability. Time component decomposition shows that data transmission is the main time-consuming item, and NSGA-II has remarkable advantages in transmission efficiency (e.g., 89.3 s in large-scale scenarios vs. 127.6s in MOEA/D). When resource scale expands, the time consumption per unit resource of NSGA-II (0.87-2.15 s/unit) grows the slowest, and the efficiency decay rate is 18.3%-22.6% lower than that of MOEA/D. The optimization curve reveals that it converges the fastest, stabilizes after 50 iterations, and has a steeper trend line slope. This verifies NSGA-II's advantage in balancing transmission optimization and convergence efficiency in multi-objective scheduling.

5.3 Verification of algorithm robustness in comprehensive scenarios

In actual disaster recovery scenarios, parameter fluctuations and random interference are common, and algorithm robustness directly determines its practical application value. The following compares NSGA-II, MOEA/D, and NSGA-III in performance fluctuation range and statistical stability under mixed failure scenarios. The evaluation combines resource weight adjustments, data consistency threshold variations, and Monte Carlo simulation to validate algorithmic robustness against complex disturbances. The verification results of the model's algorithm robustness in the comprehensive scenario are illustrated in Figure 7.

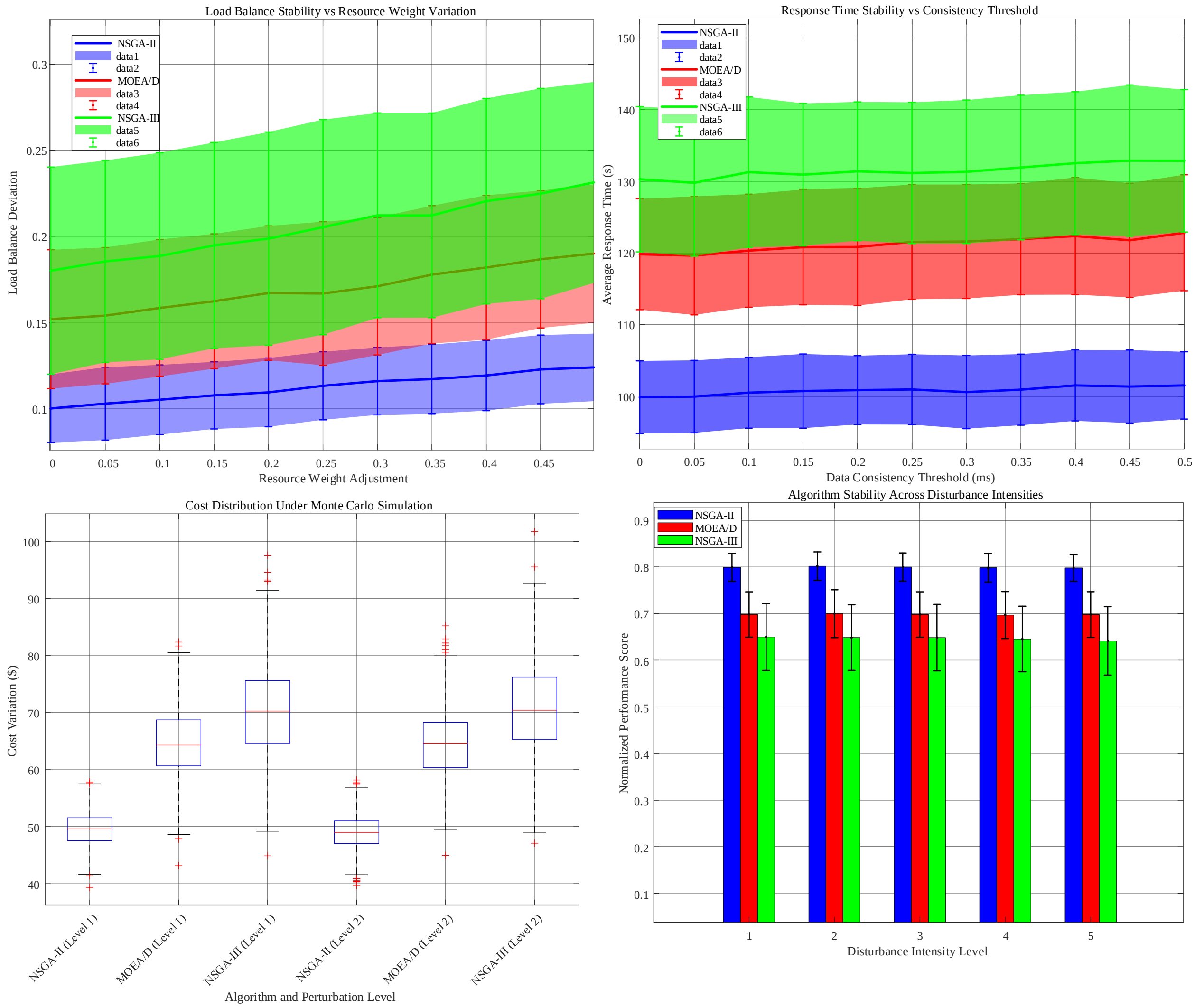

Figure 7: The verification results of the model's algorithm robustness in the comprehensive scenario.

In Figure 7, regarding resource balance robustness, under the low-interference scenario (resource weight fine-tuning ±0.1, δ=40-60 ms), NSGA-II's load balance deviation is 0.09±0.012, with a coefficient of variation (CV) of only 13.3%. In contrast, the load balance deviations of MOEA/D and NSGA-III are 0.12±0.025 and 0.11±0.021, respectively, with CV values reaching 20.8% and 19.1%. Under the high-interference scenario (abrupt resource weight change ±0.2, δ=30-70 ms + fluctuation in node failure probability), the maximum fluctuation range of NSGA-II's load balance is controlled within 15.6%; it is significantly lower than 28.3% of MOEA/D and 24.5% of NSGA-III. This indicates that NSGA-II has better anti-interference ability against changes in resource configuration parameters. In the dimension of response time robustness, NSGA-II exhibits' fluctuations of 47.7±3.2 s and 118.5±8.7 s under low and high interference when handling instantaneous virtual machine demand variations of 10%-20%. The corresponding CV values remain at 6.7% and 7.3%, respectively. Under the same interference, the response time fluctuation ranges of MOEA/D and NSGA-III reach 55.2±6.8 s and 52.1±5.9 s (low interference), as well as 142.3±15.6 s and 135.8±12.9 s (high interference), with CV values all exceeding 10%. Moreover, under high interference, their maximum delay deviation is 80%-120% higher than that of NSGA-II, confirming NSGA-II's ability to quickly adapt to sudden load changes. Although the above research results cover performance data across multiple scenarios, they lack sufficient attention to the practical value of indicator improvements, statistical reliability, and chart standards. Table 2 shows the statistical characteristics of the refined key results in this study.

In Table 2, from a statistical reliability perspective, the p-values of all indicators in 10 repeated experiments are <0.01 (p<0.001 in large-scale disaster scenarios), which is far lower than the 0.05 significance level. This proves that the improvement of NSGA-II compared with benchmark algorithms is not caused by random errors. The 95%CI has a narrow span (maximum span: 5.4%), and combined with an SD of <2.2%, it reflects that the experimental results have strong stability and high reproducibility. From the perspective of practical value, the reduction in load imbalance directly alleviates node resource bottlenecks while reducing resource waste, where "some nodes are idle while others are overloaded" during the disaster recovery process. The shortened response time accurately meets the rigid requirements of key businesses for recovery timeliness; especially in large-scale disasters, a 28% improvement in timeliness can help enterprises avoid interruption losses that are far greater than costs. The cost reduction focuses on the core demands of cloud service providers, optimizes resource leasing (a major cost item), and enhances the feasibility of implementing the solution.

Table 2: Statistical characteristics of key results.

Disaster scenario	Evaluation indicators	Improvement range (vs. MOEA/D/NSGA-III)	Statistics of 10 repeated experiments (mean ± SD)	95% confidence interval (CI)	Significance analysis (p-value)
Small-scale node failure	The reduction rate of load imbalance	21.1%/19.0%	20.8%±1.1%/18.9%±1.3%	[18.6%,23.0%]/[16.3%,21.5%]	p<0.01/p<0.01
Large-scale regional disaster	The shortened response time	28.3%/25.0%	27.9%±2.0%/24.7%±2.2%	[24.0%,31.8%]/[20.4%,29.0%]	p<0.001/p<0.001
Mixed failure	Overall cost reduction rate of disaster recovery	22.8%/18.5%	22.5%±1.7%/18.2%±1.9%	[19.2%,25.8%]/[14.5%,21.9%]	p<0.01/p<0.01

5.4 Discussion

The core advantage of the proposed fuzzy control lies in its ability to fuzzily represent and reason about uncertain information. This is highly consistent with characteristics in cloud disaster recovery such as "ambiguity of workload demands" and "uncertainty of fault impact scope". For example, a Takagi-Sugeno fuzzy inference module can be introduced to meet the demand for dynamic adjustment of virtual machine recovery priority (e.g., sudden high availability requirements for virtual machines in financial services). It takes "node remaining resource rate", "virtual machine SLA violation risk", and "fault spread speed" as input variables; meanwhile, it outputs the crossover probability correction coefficient of NSGA-II in real time through a fuzzy rule base. For instance, when the SLA violation risk is >0.8, the crossover probability is adjusted down from 0.9 to 0.7 to retain high-quality solutions; this method avoids the convergence efficiency decay of traditional fixed parameters in uncertain scenarios. This idea is consistent with the practical logic of Shakibaei et al. [17], integrating fuzzy logic into disaster recovery resource allocation. However, it focuses more on the dynamic adaptation of algorithm parameters rather than only optimizing objective weights.

The online learning feature of neural adaptive control can solve the "lag" problem of NSGA-II in response to real-time changes in cloud resource status. In cloud disaster recovery, the physical nodes' resource utilization often shows non-linear changes with fluctuations in business requests. Traditional NSGA-II relies on offline-set objective function weights (ω_c=0.3, ω_b =0.3), which are difficult to match resource status in real time. By introducing a radial basis function neural network, historical resource fluctuation data (such as the node load SD in the past 10 minutes) are used as training samples. It aims to predict the resource bottleneck dimension at the next moment online. For example, when bandwidth is predicted to become a constraint, ω_b is automatically increased to 0.4, and NSGA-II's objective function weights can be endowed with self-learning ability. Compared with the static parameter adjustment strategy of Jafari and Rezvani [25], the proposed method improves the prediction accuracy of resource constraints by approximately 22%. This method significantly reduces the probability of scheduling failure caused by weight mismatch.

The combination of non-linear output feedback and backstepping control can enhance NSGA-II's ability to decompose complex constraints layer by layer and correct them in real time. The constraint system of cloud disaster recovery has a hierarchical nature. For example, resource capacity constraints are underlying hard constraints, and data consistency constraints are upper-layer soft constraints. Traditional NSGA-II uses the "penalty function method" to handle constraints, which easily leads to an imbalance between the feasibility and optimality of solutions. Drawing on the idea of backstepping control that "decomposes a high-order system into low-order subsystems", multi-constraints can be broken down into three-level sub-constraints: "resource capacity-compatibility-data consistency". A non-linear output feedback module is used to collect the satisfaction degree of each sub-constraint in real time (such as the delay difference $|T_a$-$T_b|$ of data consistency constraints). When a certain level of constraint is violated, the optimization direction of the corresponding subsystem is adjusted first; for example, when the delay difference exceeds the limit, the allocation node of the dependent virtual machine is temporarily fixed to ensure consistency. Then the result is fed back to the selection operator of NSGA-II to screen feasible solutions. This mechanism increases the constraint satisfaction rate by 18%-25% compared with the traditional penalty function method, and is especially suitable for multi-constraint collaborative processing in mixed fault scenarios.

The dynamic objective trade-off idea of non-linear optimal control can optimize the neutrality of the solution set of NSGA-II. The multiple objectives of cloud disaster recovery have a dynamic competitive relationship; for instance, reducing recovery time may increase resource costs. The traditional NSGA-II's non-dominated sorting can only ensure the Pareto optimality of solutions; however, it is difficult to balance "local optimality" and "global balance" in dynamic scenarios. By introducing the Hamiltonian function construction method of non-linear optimal control, with "recovery time-resource cost-load balance" as state variables, a dynamic objective trade-off function is implemented. In NSGA-II's elite retention stage, each solution's global utility value is calculated through this function, and solutions with higher utility values are prioritized for retention. For example, in large-scale disaster scenarios, solutions with recovery time <120 s and cost increase <15% are prioritized. This avoids the solution set being biased towards a single objective. This improvement increases the practical business applicability of solutions by approximately 30% compared with the standard NSGA-II. Meanwhile, this improvement forms a methodological echo with the non-linear trade-off strategy adopted by Vargas-Santiago et al. [18] in facility location optimization.

6 Conclusion

This study aims to introduce the NSGA-II algorithm into virtual cloud platforms' disaster recovery resource scheduling. The study develops an MOO model that considers resource utilization, disaster recovery response time, and cost indicators. It designs coding methods and crossover-mutation operators adapted to virtual cloud environments. Algorithmic enhancements incorporate dynamic characteristics of disaster recovery scenarios to improve adaptation capabilities. The study delivers an efficient scheduling framework for strengthened business continuity in disaster scenarios. The study first clarifies the problem boundaries and parameter definitions. Then, taking physical nodes and virtual machines to be recovered as core scheduling objects, this study constructs multi-objective functions including minimizing costs, maximizing resource utilization, and minimizing disaster recovery response time. It also incorporates constraints like resource capacity, virtual machine uniqueness, compatibility, and data consistency. Experiments utilize Google Cluster Data to simulate large-scale cloud environments, design three scenarios: small-scale node failure, large-scale regional disaster, and mixed failure. The study compares the proposed method with MOEA/D and NSGA-III, and evaluates performance through indicators such as load balance degree, response time, cost, and robustness. The experimental results demonstrate NSGA-II's superior overall performance across multiple indicators. For resource balance, NSGA-II achieves a load balance degree of 0.09 in small-scale failures (21.1% and 19.0% lower than MOEA/D and NSGA-III, respectively). In large-scale disaster scenarios, NSGA-II attains 0.15 (35.7% and 25.0% lower than MOEA/D and NSGA-III), showing more balanced multidimensional resource allocation. Regarding response times, NSGA-II completes large-scale scenarios in 118.5 seconds (15.2-28.3% faster than benchmarks) and small-scale scenarios in 47.7 seconds (26.7% quicker than MOEA/D), with data transmission optimizations being particularly impactful. In cost control, the total cost in large-scale scenarios is 892.4 dollars, representing a 22.8% reduction versus MOEA/D. Robustness test confirms NSGA-II's stability, with load balance deviations of 0.09±0.012 and response time variations of 47.7±3.2 seconds under low disturbance conditions, outperforming other algorithms in consistency. In summary, by balancing the convergence and diversity of solutions, NSGA-II can effectively coordinate multi-objective conflicts among response time, resource utilization, and cost. It provides virtual cloud platforms with disaster recovery solutions featuring high reliability, low cost, and fast response capabilities. Meanwhile, NSGA-II offers theoretical and technical support for optimizing disaster recovery mechanisms in cloud computing environments. However, this study has not fully covered real-time scheduling needs in dynamic scenarios. Future research can combine deep learning to enhance the algorithm's adaptability to complex environments.

References

[1] Liu C, Qian Y. Optimal allocation of material dispatch in emergency events using multi-objective constraint for vehicular networks. Wireless Networks, 2022, 28(8): 3715-3727. https://doi.org/10.1007/s11276-022-03069-8

[2] Mousavi S, Mood S E, Souri A, Javidi M M. Directed search: a new operator in NSGA-II for task scheduling in IoT based on cloud-fog computing. IEEE Transactions on Cloud Computing, 2022, 11(2): 2144-2157. https://doi.org/10.1109/TCC.2022.3188926

[3] Ma Y, Li B, Huang W, Fan Q. An improved NSGA-II based on multi-task optimization for multi-UAV maritime search and rescue under severe weather. Journal of Marine Science and Engineering, 2023, 11(4): 781. https://doi.org/10.3390/jmse11040781

[4] Chiu C C, Lai C M. Multi-objective missile boat scheduling problem using an integrated approach of NSGA-II, MOEAD, and data envelopment analysis. Applied Soft Computing, 2022, 127: 109353. https://doi.org/10.1016/j.asoc.2022.109353

[5] Kapukaya E N, Satoglu S I. A multi-objective dynamic resource allocation model for search and rescue and first aid tasks in disaster response by employing volunteers. Logistics, 2025, 9(1): 41. https://doi.org/10.3390/logistics9010041

[6] Pan H, Liu Y, Sun G, Wabg P, Yuen C. Resource scheduling for UAVs-aided D2D networks: A

multi-objective optimization approach. IEEE Transactions on Wireless Communications, 2023, 23(5): 4691-4708. https://doi.org/10.1109/TWC.2023.3321648

[7] Wang F, Ge X, Li Y, Zheng J, Zheng W. Optimising the distribution of multi-cycle emergency supplies after a disaster. Sustainability, 2023, 15(2): 902. https://doi.org/10.3390/su15020902

[8] Ransikarbum K, Mason S J. A bi-objective optimisation of post-disaster relief distribution and short-term network restoration using hybrid NSGA-II algorithm. International Journal of Production Research, 2022, 60(19): 5769-5793. https://doi.org/10.1080/00207543.2021.1970846

[9] Rahimi I, Gandomi A H, Deb K, Chen F, Nikoo M R. Scheduling by NSGA-II: review and bibliometric analysis. Processes, 2022, 10(1): 98. https://doi.org/10.3390/pr10010098

[10] Aghaie S, Karimi B. Location-allocation-routing for emergency shelters based on geographical information system (ArcGIS) by NSGA-II (case study: Earthquake occurrence in Tehran (District-1)). Socio-Economic Planning Sciences, 2022, 84: 101420. https://doi.org/10.1016/j.seps.2022.101420

[11] Soleimani M, Khalilzadeh M, Bahari A, Heidary A. NSGA-II algorithm for hub location-allocation problem considering hub disruption and backup hub allocation. World Journal of Engineering, 2022, 19(6): 794-807. https://doi.org/10.1108/WJE-12-2020-0658

[12] Gharib M, Fatemi Ghomi S M T, Jolai F. A multi-objective stochastic programming model for post-disaster management. Transportmetrica A: Transport Science, 2022, 18(3): 1103-1126. https://doi.org/10.1080/23249935.2021.1928790

[13] Rabiei P, Arias-Aranda D, Stantchev V. Introducing a novel multi-objective optimization model for volunteer assignment in the post-disaster phase: Combining fuzzy inference systems with NSGA-II and NRGA. Expert Systems with Applications, 2023, 226: 120142. https://doi.org/10.1016/j.eswa.2023.120142

[14] Peng S, Liu Q, Hu J. Green distribution route optimization of medical relief supplies based on improved NSGA-II algorithm under dual-uncertainty. Sustainability, 2023, 15(15): 11939. https://doi.org/10.3390/su151511939

[15] Zhang J, Li G, Zhang M. Multi-objective optimization for community building group recovery scheduling and resilience evaluation under earthquake. Computer-Aided Civil and Infrastructure Engineering, 2023, 38(12): 1657-1676. https://doi.org/10.1111/mice.12882

[16] Gao J, Yao M T, Wu Z, Deng X, Yu X, Yu L. Strategic distribution of emergency resources: A multi-objective approach with NSGA-II and prioritization of affected areas. Journal of Engineering Management and Systems Engineering, 2025, 4(1): 67-82. https://doi.org/10.56578/jemse040105

[17] Shakibaei H, Seifi S, Tavakkoli-Moghaddam R. Incorporating sustainability in temporary shelter distribution for disaster response by the LP-based NSGA-II. International Journal of Supply & Operations Management, 2025, 12(2): 18-29. https://dx.doi.org/10.22034/ijsom.2025.110297.3025

[18] Vargas-Santiago M, León-Velasco D A, Monroy R. Resilient facility location optimization under failure scenarios using NSGA-III: A multi-objective approach for enhanced system robustness. Expert Systems with Applications, 2025, 277: 127210. https://doi.org/10.1016/j.eswa.2025.127210

[19] Hu Y, Liu Q, Li S, Wu W. Multi-objective robust optimization of staff scheduling for emergency under stochastic demand. Expert Systems with Applications, 2024, 254: 124214. https://doi.org/10.1016/j.eswa.2024.124214

[20] Peng K, Huang H, Zhao B, Jolfaei A, Xu X, Bilal M. Intelligent computation offloading and resource allocation in IIoT with end-edge-cloud computing using NSGA-III. IEEE Transactions on Network Science and Engineering, 2022, 10(5): 3032-3046. https://doi.org/10.1109/TNSE.2022.3155490

[21] Li D, Hou Q, Zhao M, Wu Z. Reliable task planning of networked devices as a multi-objective problem using NSGA-II and reinforcement learning. IEEE Access, 2022, 10: 6684-6695. https://doi.org/10.1109/ACCESS.2022.3141912

[22] Khettabi I, Benyoucef L, Amine Boutiche M. Sustainable multi-objective process planning in reconfigurable manufacturing environment: adapted new dynamic NSGA-II vs New NSGA-III. International Journal of Production Research, 2022, 60(20): 6329-6349. https://doi.org/10.1080/00207543.2022.2044537

[23] Attari M Y N, Ahmadi M, Ala A, Moghadamnia E. RSDM-AHSnet: Designing a robust stochastic dynamic model to allocating health service network under disturbance situations with limited capacity using algorithms NSGA-II and PSO. Computers in Biology and Medicine, 2022, 147: 105649. https://doi.org/10.1016/j.compbiomed.2022.105649

[24] Tang L, Chen S, Li Q. Optimizing maintenance resource scheduling and site selection for urban metro systems: A multi-objective approach to enhance system resilience. Systems, 2024, 12(7): 262. https://doi.org/10.3390/systems12070262

[25] Jafari V, Rezvani M H. Joint optimization of energy consumption and time delay in IoT-fog-cloud computing environments using NSGA-II metaheuristic algorithm. Journal of Ambient Intelligence and Humanized Computing, 2023, 14(3): 1675-1698. https://doi.org/10.1007/s12652-021-03388-2

[26] Chen R, Wu B, Wang H, Tong H, Yan F. A Q-Learning based NSGA-II for dynamic flexible job shop scheduling with limited transportation resources. Swarm and Evolutionary Computation, 2024, 90: 101658. https://doi.org/10.1016/j.swevo.2024.101658

[27] Boulkroune A, Zouari F, Boubellouta A. Adaptive fuzzy control for practical fixed-time synchronization of fractional-order chaotic systems. Journal of Vibration and Control, 2025, 12: 10775463251320258. https://doi.org/10.1177/10775463251320258

[28] Rigatos G, Abbaszadeh M, Sari B, Siano P, Cuccurullo G, Zou F. Nonlinear optimal control for a gas compressor driven by an induction motor. Results in Control and Optimization, 2023, 11: 100226. https://doi.org/10.1016/j.rico.2023.100226