Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.081803

ARTICLE

Adaptive Pareto-Based Multi-Agent Decision Model for Resource Management in Cloud Business Intelligence Systems

Khamza Eshankulov[1], Bahodir Muminov[2], Robiya Farmonova[1], Dilnavoz Sodikova[3], Bakhriddin Bozorov[4], Zavqiddin Temirov[5] and Rashid Nasimov[6,*]

[1]Department of Applied Mathematics and Programming Technologies, Bukhara State University, Bukhara, Uzbekistan
[2]Department of Artificial Intelligence, Tashkent State University of Economics, Tashkent, Uzbekistan
[3]Department of Biomedical Engineering, Biophysics and Informatics, Bukhara State Medical Institute, Bukhara, Uzbekistan
[4]Chief Innovation Officer, "Navoi Mining and Metallurgical Company" Joint-Stock Company, Navoi, Uzbekistan
[5]Department of Digital Technologies, Alfraganus University, Yukori Karakamish Street 2a, Tashkent, Uzbekistan
[6]Department of Computer Engineering, Gachon University, Seongnam-Si, Republic of Korea

*Corresponding Author: Rashid Nasimov. Email: rashid.nasimov@tsue.uz

Received: 15 March 2026; Accepted: 05 June 2026; Published: 23 July 2026

ABSTRACT: Cloud-based Business Intelligence (BI) systems operate under highly dynamic analytical workloads, including bursty OLAP queries, concurrent aggregations, and real-time microservice interactions, where static resource allocation leads to latency spikes and inefficient resource utilization. This paper proposes a decentralized adaptive Pareto-based multi-agent decision model for real-time resource coordination in cloud BI microservice environments. The agent placement problem is formulated as a multi-criteria decision process that minimizes service response latency, improves computational resource utilization, and preserves Quality-of-Service (QoS) stability. Instead of constructing a centralized global optimization policy, the proposed framework relies on decentralized locally Pareto-efficient decisions combined with adaptive priority regulation driven by QoS deviation. The approach is evaluated through large-scale controlled simulation and validated in a Kubernetes-based pilot cloud environment. Experimental results demonstrate up to 54% latency reduction compared to static allocation and 22% improvement over GA-based optimization, with enhanced CPU utilization balance under dynamic workloads. Statistical analysis confirms the significance of improvements ($p < 0.05$). The proposed model ensures bounded monotonic decision transitions without centralized orchestration or predictive training, making it suitable for real-time cloud-native BI service ecosystems.

KEYWORDS: Cloud resource allocation; pareto decision making; decentralized coordination; business intelligence microservices; adaptive multi-agent systems

1 Introduction

In recent years, decision-making processes in modern organizations and higher education institutions have increasingly relied on large-scale, heterogeneous, and continuously updated data streams. In this context, BI systems have become a widely adopted information technology for analyzing organizational activities and supporting both strategic and operational decision-making [1,2].

However, traditional BI systems face significant limitations when dealing with large volumes of dynamically changing data, which has led to a growing trend toward migrating BI platforms to cloud computing environments. As a result, cloud-based BI systems have substantially expanded the possibilities for scalability, flexibility, and efficient utilization of computational resources [3].

To address these challenges, multi-agent systems have emerged as a promising approach. In a multi-agent environment, each agent operates as an autonomous decision-making entity capable of monitoring resource states, performing local analysis, and generating adaptive control decisions [4,5]. Deploying agents within service-oriented and microservice-based architectures in cloud infrastructures enhances the scalability and reliability of BI systems [6,7].

Resource allocation in cloud-based BI systems involves multiple conflicting objectives—minimizing latency, maximizing utilization, and preserving service quality—that cannot be optimized independently without trade-offs [8–10]. Pareto-based optimization provides a principled mechanism for identifying balanced solutions across such competing criteria [11,12].

In this paper, we propose a multi-agent resource allocation model for cloud-based BI systems, in which the agent placement process is formally modeled as a multi-criteria Pareto-based optimization problem. The proposed approach is designed to support the dynamic real-time placement of agents under service-oriented and microservice-based architectures, aiming to improve resource utilization efficiency while reducing service response latency. In this way, the present study introduces a complementary and extensible scientific solution that advances the state of the art in cloud-based BI systems.

Although resource allocation and optimization in cloud environments have been widely studied, multi-agent and Pareto-based approaches specifically tailored to BI systems have not been sufficiently analyzed. Existing studies often rely on centralized management and ignore BI-specific features such as real-time and multi-service execution. In the proposed approach, each agent makes independent decisions based on the current system state and dynamically adjusts its priorities through local QoS deviations.

Existing cloud resource allocation methods largely rely on centralized schedulers or reinforcement learning models requiring global synchronization and computationally expensive retraining procedures. Such approaches become inefficient in cloud-based BI environments characterized by bursty OLAP workloads and dynamically changing microservice interactions. In contrast, the proposed framework enables decentralized Pareto-consistent adaptive coordination without centralized orchestration.

Despite the extensive research on cloud resource allocation and multi-agent optimization, existing approaches do not explicitly address the dynamic and analytical nature of BI workloads, where multiple services interact in real time and require continuous multi-criteria adaptation. In particular, the lack of decentralized decision mechanisms with adaptive QoS-driven regulation remains an open challenge. To address this gap, this paper proposes a novel adaptive Pareto-based multi-agent decision model tailored for cloud-based BI environments.

Contributions of this Paper. This paper makes the following contributions:

1. We reformulate cloud BI resource coordination as a decentralized Pareto-consistent adaptive decision process operating without centralized optimization or global retraining.
2. We propose a QoS-driven adaptive priority regulation mechanism, in which each agent dynamically updates decision weights based on real-time deviation between target and observed QoS metrics.
3. We introduce a Pareto-based local decision framework that ensures only Pareto-improving or Pareto-preserving transitions are accepted, maintaining multi-criteria balance between latency, resource utilization, and service stability.
4. We provide a stability-oriented analysis showing convergence to a locally stable Pareto-efficient operating region under bounded workload variations.
5. We validate the proposed approach through simulation and a Kubernetes-based pilot cloud environment.

Unlike existing centralized or RL-based cloud allocation approaches, the proposed framework enables lightweight decentralized Pareto-consistent adaptive coordination tailored for dynamic BI microservice environments without requiring global optimization or offline policy retraining.

2 Related Work

2.1 Cloud-Based BI Systems

In recent years, BI systems have increasingly migrated to cloud environments to leverage elastic computing resources and support large-scale analytical workloads [13–15]. Cloud-based BI platforms implement analytical functionalities as independent microservices, enabling scalable and flexible data processing [16,17]. However, existing approaches predominantly rely on centralized resource management strategies that do not fully account for the distributed, real-time, and multi-service execution characteristics of cloud-based BI environments.

2.2 Resource Allocation in Cloud Environments

Resource allocation in cloud computing environments aims to ensure the efficient utilization of computational power, memory, and network resources. Existing studies have investigated this problem through a variety of optimization approaches, ranging from single-objective heuristics to more complex multi-criteria methods [18–20]. In practice, resource allocation in cloud computing environments is inherently a multi-criteria problem that requires balancing multiple, often conflicting objectives. Consequently, recent studies have increasingly adopted Pareto-based multi-criteria optimization approaches as a promising solution for handling such complex decision-making problems. However, existing Pareto-front-based approaches primarily focus on general-purpose computing tasks or infrastructure-level optimization, while insufficiently considering the analytical workloads and service-oriented execution characteristics that are specific to BI systems [21]. In addition, the network function virtualization (NFV) domain has extensively studied service placement and resource allocation problems in distributed cloud environments [22,23]. Unlike NFV-based approaches, which often rely on centralized orchestration and predefined service chains, the proposed model focuses on decentralized agent-based decision-making tailored for dynamic BI workloads.

Although existing cloud resource allocation approaches demonstrate effective optimization performance, many of them rely on centralized coordination, predefined weighting schemes, or computationally intensive optimization procedures. Such limitations reduce adaptability in dynamic BI environments with fluctuating analytical workloads and heterogeneous service interactions.

2.3 Multi-Agent Optimization Approaches

Multi-agent systems have been widely adopted as an effective approach for automating decision-making processes in complex and distributed environments [24]. In addition, formal modeling techniques such as Petri nets have been applied to represent industrial workflows and interactions between system components in distributed environments. In the context of cloud computing, such approaches are actively employed to support resource monitoring, workload balancing, and coordination among services [25]. The ability of agents to make autonomous decisions based on local information contributes to improved overall system efficiency [26]. Table 1 summarizes the key differences between existing resource management approaches in cloud computing environments and the model proposed in this study.

Table 1: Comparative analysis of existing cloud resource allocation approaches and the proposed decentralized Pareto-based multi-agent model.

Approach	Control Type	Optimization Type	BI-Oriented	Real-Time Support	Real Cloud Environment
Conventional cloud resource management	Centralized	Single-objective	Not supported	Limited	Not supported
Classical Pareto-based optimization	Centralized	Multi-objective (offline)	Not supported	Not supported	Limited
Multi-agent approaches	Partially decentralized	Single-objective	Limited	Limited	Not supported
Proposed Pareto-based multi-agent model	Decentralized	Pareto-based (real-time)	Full	Full	Full

As shown in Table 1, most existing approaches are primarily oriented toward general cloud computing tasks and do not sufficiently account for BI-specific real-time analytical processes and microservice-based environments. In contrast, the proposed approach differs from prior work by adopting a decentralized control principle, employing Pareto-based multi-criteria optimization, and being experimentally validated in a real cloud environment. Although the reviewed multi-agent optimization approaches demonstrate effectiveness in distributed settings, many of them rely on a single optimization objective or require centralized coordination mechanisms, which limit their applicability in dynamic, multi-service, real-time BI environments.

To reduce potential bias in the selection of related work, a structured literature review (SLR) approach based on the PICOC framework (Population, Intervention, Comparison, Outcome, Context) was employed [27]. Relevant studies were systematically retrieved from IEEE Xplore, Scopus, and SpringerLink using the following predefined search string: ((“cloud resource allocation” OR “multi-agent resource allocation” OR “cloud resource management”) AND (“multi-agent systems” OR “autonomous agents” OR “agent-based”) AND (“Pareto optimization” OR “multi-objective optimization” OR “Pareto-based”)). Studies were included if they were peer-reviewed, published in English between 2015 and 2024, and directly addressed cloud resource allocation, multi-agent coordination, or Pareto-based optimization. Studies were excluded if they were unrelated to cloud computing, unavailable in full text, or published as theses or technical reports. After applying these criteria, 14 relevant studies were selected and compared in Table 1.

Unlike recent decentralized multi-agent resource allocation approaches that rely on scalarization or predefined weighting strategies, the proposed model adopts an adaptive Pareto-based decision mechanism without fixed global objective aggregation. Existing methods often depend on static priority configurations or centralized coordination, limiting responsiveness in dynamic BI workloads. In contrast, the proposed framework enables decentralized decision-making with adaptive priority regulation based on QoS deviations, supporting more flexible and real-time resource coordination.

Existing RL-based cloud schedulers often require iterative policy training and global reward propagation, which may limit responsiveness under rapidly changing BI analytical workloads. In contrast, the proposed framework employs lightweight decentralized adaptation without offline training procedures.

3 System Architecture and Problem Definition

This section presents the overall architecture proposed for cloud-based Business Intelligence (BI) systems and describes the multi-agent model designed for resource management. In addition, the resource allocation process is formally formulated as a multi-criteria Pareto-optimal optimization problem. The proposed methodology is developed by explicitly considering BI-specific characteristics, including service-oriented execution, distributed system architecture, and real-time analytical requirements.

3.1 Cloud-Based BI Architecture

The proposed cloud-based BI architecture is built upon service-oriented and microservice design principles, aiming to effectively exploit the elastic capabilities of cloud computing infrastructures. The architecture consists of multiple logical layers that collectively support data acquisition, storage, analytical processing, and the delivery of analytical results to end users (Fig. 1).

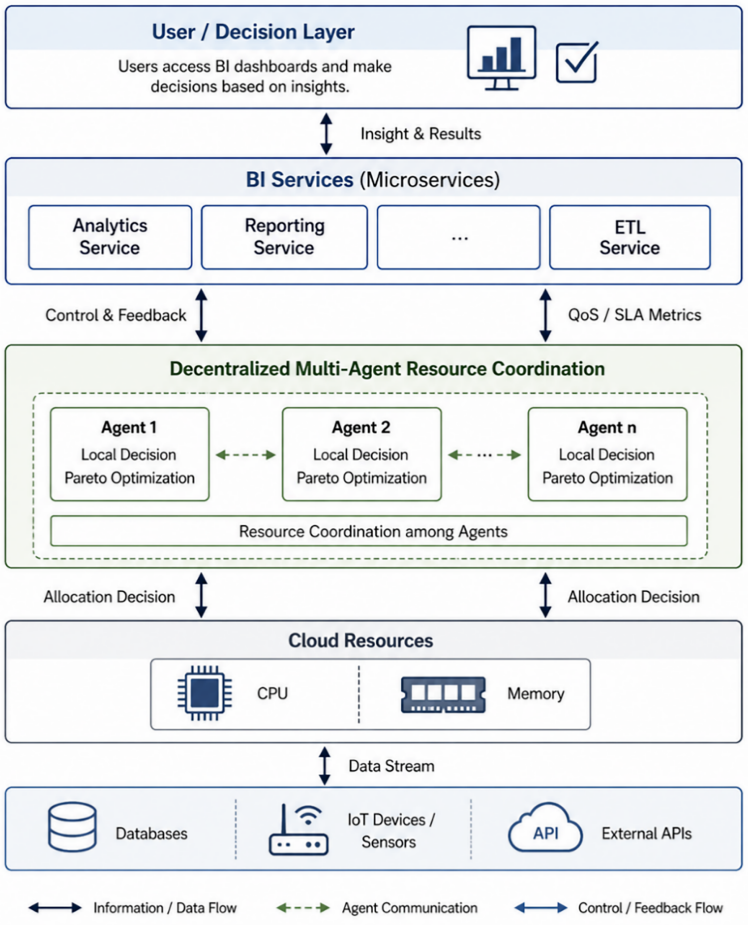


Figure 1: Architecture of the proposed cloud-based BI system illustrating microservice-based BI ser-vices and a decentralized multi-agent resource management layer with local Pareto-based decision making guided by QoS/SLA feedback.

The architecture spans three layers: IaaS (virtual machines and containers), PaaS (service deployment and scaling), and an application layer comprising BI microservices such as OLAP, reporting, and visualization modules.

3.2 Multi-Agent Resource Model

In the cloud-based BI environment, the resource management process is organized using a multi-agent system. Each agent is considered an autonomous decision-making entity responsible for monitoring, analyzing, and managing resource demands for a specific BI service or a group of services. Agents make decisions based on local information and do not require complete global knowledge of the overall system state.

Formally, the set of agents in the cloud-based BI system is defined as follows:

$$A = \{a_1, a_2, \ldots, a_n\} \tag{1}$$

where each a_i agent corresponds to a specific BI service or analytical process.

The set of available computational resources in the cloud environment is defined as follows:

$$R = \{r_1, r_2, \ldots, r_m\} \tag{2}$$

where r_j denotes a computational resource such as processing capacity, memory, or network bandwidth.

The interaction model between agents and computational resources in the cloud-based BI environment is illustrated in Fig. 2, showing the assignment of agents to BI services, resource state monitoring, and their interactions during the resource allocation process.

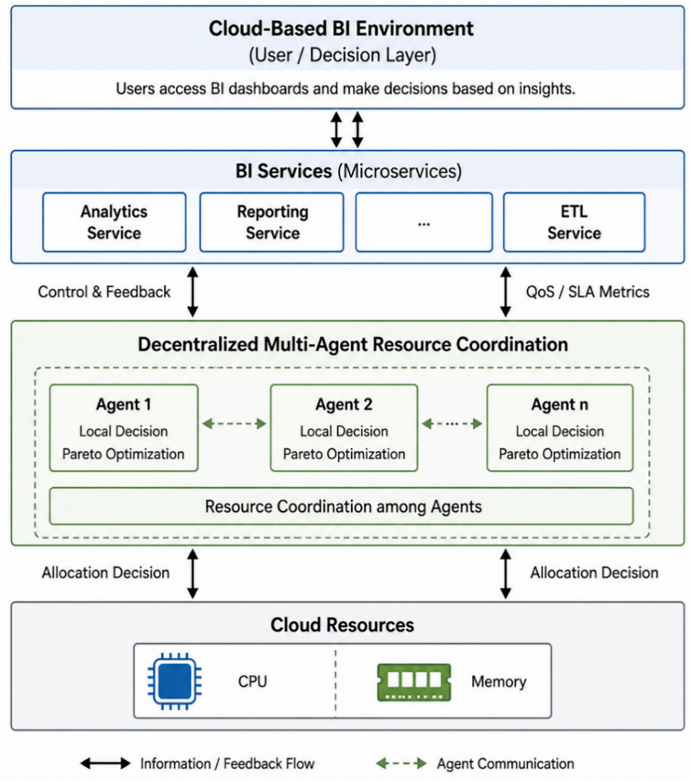


Figure 2: Multi-agent resource interaction model for resource monitoring and allocation in a cloud-based BI environment.

As illustrated in Fig. 2, agents monitor computational resources, make decisions, and adaptively manage their placement. In the proposed model, coordination is achieved indirectly through the shared cloud environment rather than direct communication, reducing reliance on centralized control. Each agent evaluates its BI service based on allocated resources and independently optimizes system efficiency, enabling a scalable and flexible response to dynamic analytical workloads.

3.3 Pareto-Optimal Problem Formulation

In cloud-based BI systems, resource allocation requires simultaneously minimizing response latency, maximizing resource utilization, and maintaining service stability criteria that are often conflicting. Optimizing them independently leads to suboptimal decisions. Therefore, the problem is formally defined within a multi-criteria Pareto-optimal optimization framework.

To formally represent this problem, a utility function is defined for each agent α_i as follows:

$$U_i = \omega_1 \cdot (1/L_i) + \omega_2 \cdot R_i - \omega_3 \cdot C_i \tag{3}$$

where L_i denotes the response latency of BI services, R_i represents the efficiency of computational resource utilization, and C_i represents the service stability (QoS level) of the BI service. The weighting coefficients ω_1, ω_2, ω_3 reflect the relative importance of each criterion and satisfy the constraint. The linear form of the utility function was selected for its computational tractability and interpretability in real-time decision-making contexts. While nonlinear aggregation functions may capture more complex preference structures, they introduce additional computational overhead that conflicts with the real-time operational requirements of cloud-based BI environments. The linear scalarization is applied exclusively as a preference selector over the Pareto-efficient candidate set, not as a global optimization objective a distinction that preserves the multi-criteria consistency of the framework.

$\omega_1 + \omega_2 + \omega_3 = 1.$

To ensure comparability among heterogeneous criteria and prevent scale dominance, all objective components are normalized to the interval $[0,1]$.

$L_i \in [0,1]$, $R_i \in [0,1]$, and $C_i \in [0,1]$

This normalization prevents scale dominance among criteria and stabilizes adaptive weight updates under dynamic workload conditions. The linear utility function is employed not as a global optimization objective, but as a preference function defined over the Pareto-efficient solution set. This design preserves multi-criteria consistency while enabling adaptive and context-aware decision selection. To enable dynamic adaptation of decision priorities based on real-time system feedback, the weight update rule for each agent is defined as follows:

$$w_i(t+1) = w_i(t) + \alpha \left(Q_i{}^{target} - Q_i{}^{observed}(t) \right) \tag{4}$$

the parameter $\alpha \in (0, 1]$ denotes the adaptation rate that controls the speed of priority adjustment. $Q_i{}^{target}$ represents the desired Quality-of-Service level for criterion i, while $Q_i{}^{observed}(t)$ is the measured QoS value at iteration t.

After each update, the weight vector is constrained to remain non-negative and normalized, such that the sum of all weights equals one. This ensures that the relative importance of decision criteria remains bounded and interpretable throughout the iterative process.

$\sum_j w_j = 1 \, and \, w_j \geq 0 \, for \, all \, criteria \; j.$

This constraint prevents uncontrolled growth of weights and guarantees stability of the adaptive decision-making mechanism under dynamic workload conditions.

The update rule increases the weight of criteria whose performance degrades and decreases the weight of satisfied criteria, allowing the decision policy to adapt to the current operational state of the system.

Definition 1 (Allocation State Space): *Let S denotes the finite set of feasible allocation states satisfying the cloud resource capacity constraints. Each state $s \in S$ corresponds to a specific assignment of agents to computational and memory resources within the bounded infrastructure. Since resource capacities are finite and allocations are discrete, the set S is finite.*

Lemma 1 (Weight Invariance): *Under the adaptive weight update mechanism introduced above, the weights remain non-negative and normalized at every iteration. That is*,

$$\begin{aligned} &w_k(t) > 0 \, for \, all \, criteria \; k; \\ &\textstyle\sum_k w_k(t) = 1 \, for \, all \, iterations \, t. \\ &U_i(x) \geq U_i(y) \; \forall i \, and \, \exists j: U_j(x) \geq U_j(y). \end{aligned} \tag{5}$$

Decision x is considered Pareto-dominant over y, meaning an agent adopts a new resource allocation only if at least one criterion improves without degrading others directly integrating Pareto-optimal optimization with agents' local decision-making.

Pareto dominance induces a partial order over the feasible solution space, accepting only Pareto-improving transitions. Accordingly, the agent placement and resource allocation problem in a cloud-based BI environment is formulated as a multi-criteria optimization problem defined by the following objective vector:

$$F(x) = (f_1(x), f_2(x), f_3(x)) \tag{6}$$

where $f_1(x)$ is the function that minimizes the average response latency of BI services, $f_2(x)$ maximizes the efficiency of computational and memory resource utilization, and, $f_3(x)$ represents the function that reflects service quality (QoS) and overall system stability. These objectives are jointly aggregated within the utility function and are employed in the agents Pareto-based decision-making process.

The optimization is performed subject to the following resource constraints:

$$\textstyle\sum_{i=1^n} c_i \leq C_{total}, \sum_{i=1^n} m_i \leq M_{total} \tag{7}$$

where c_i and m_i denote the computational and memory resources allocated to agent a_i, respectively, while, C_{total} and M_{total} represent the total computational and memory resources available in the cloud infrastructure.

Proposition 1. (Stability of Utility-Guided Pareto Decisions)

Under bounded workload variations and finite resource capacity, the iterative Pareto-based decision-making process converges to a locally stable Pareto-efficient operating region.

Stability-Oriented Convergence Analysis

Convergence analysis. At iteration $t = 0$, weights are initialized with $w_k(0)>$ for all k and $\sum_k w_k(0) = 1$ by construction. Assume the invariant holds at iteration t. The update rule gives $w_k(t+1) = w_k(t) + \alpha(Q_k^{target} - Q_k^{observed}(t))$. After the update, weights are clipped to [0, 1] and renormalized by dividing by their sum,

which is strictly positive since at least one weight remains positive after clipping. Therefore $w_k(t+1) \geq 0$ for all k and $\sum_k w_k\,(t+1) = 1$. By induction, the invariant holds for all $t \geq 0$.

Let S denote the finite set of feasible allocation states under Pareto-induced partial order. Since only Pareto-improving transitions are accepted, the process forms a monotonic sequence. Given the finiteness of S and the strict partial order, infinite improving sequences and cyclic reallocations are excluded guaranteeing termination at a locally stable Pareto region. Convergence follows from two properties: finiteness of the state space and monotonicity of Pareto-consistent transitions.

Under bounded workload variations and finite resource allocation states, the proposed adaptive coordination process converges toward a locally stable Pareto-consistent allocation region. Since each adaptive update improves or preserves the local utility function while avoiding cyclic reallocations, the iterative coordination process remains stable under dynamic BI workload conditions.

Corollary 1 (Finite Convergence). The adaptive Pareto-based coordination process terminates within at most |S| iterations, where |S| denotes the cardinality of the feasible allocation state space S. This follows directly from the finiteness of S, the strict monotonicity of Pareto-consistent transitions, and the exclusion of cyclic reallocations under the induced partial order.

Remark: *The stability results hold under the assumption that workload variations remain bounded and that the resource capacity constraints defined in Eq. (7) are satisfied throughout execution..*

4 Pareto-Based Agent Placement Algorithm

This section presents a Pareto-based algorithm for agent placement and resource allocation in cloud BI environments, aimed at improving resource efficiency, reducing response latency, and maintaining system stability. The algorithm supports real-time operation by enabling dynamic resource reallocation through agent decisions based on local information.

4.1 Algorithm Description and Optimization Strategy

The proposed optimization strategy is based on Pareto dominance, where each resource allocation state is represented in a multidimensional criteria space. A solution is considered Pareto-dominant if it improves at least one criterion without degrading others, and the set of such solutions forms a balanced and feasible decision space. The algorithm integrates Pareto-based filtering with an adaptive priority mechanism, in which agents update criterion weights based on observed QoS deviations. This enables a shift from a static multi-criteria scheduler to a self-regulating distributed decision process that adapts to workload changes in real time, ensuring that each deployment decision reflects the current cloud environment. The workflow of the proposed algorithm is illustrated in Fig. 3.

Algorithm 1 presents the decentralized Pareto-based agent placement procedure used for resource coordination in cloud-based BI systems. The algorithm operates iteratively and consists of several key stages, including system state monitoring, generation of candidate allocations, multi-criteria evaluation, Pareto-based filtering, adaptive weight updating, and decision selection. At each iteration t, the global allocation state $S(t)$ is updated based on local decisions of agents. The adaptive mechanism adjusts the criteria weight vector $W(t)$ according to the deviation between target and observed QoS levels. The algorithm terminates when the convergence condition is satisfied. The convergence measure is defined as $\Delta(t) = \|S(t+1) - S(t)\|$, where $\|\cdot\|$ denotes the Euclidean norm and $\varepsilon > 0$ is a predefined convergence threshold. Convergence is achieved when $\Delta(t) < \varepsilon$, ensuring a locally stable Pareto-efficient allocation state. Since each agent operates using only local information, the approach eliminates the need for centralized control and demonstrates scalability in dynamic BI environments.

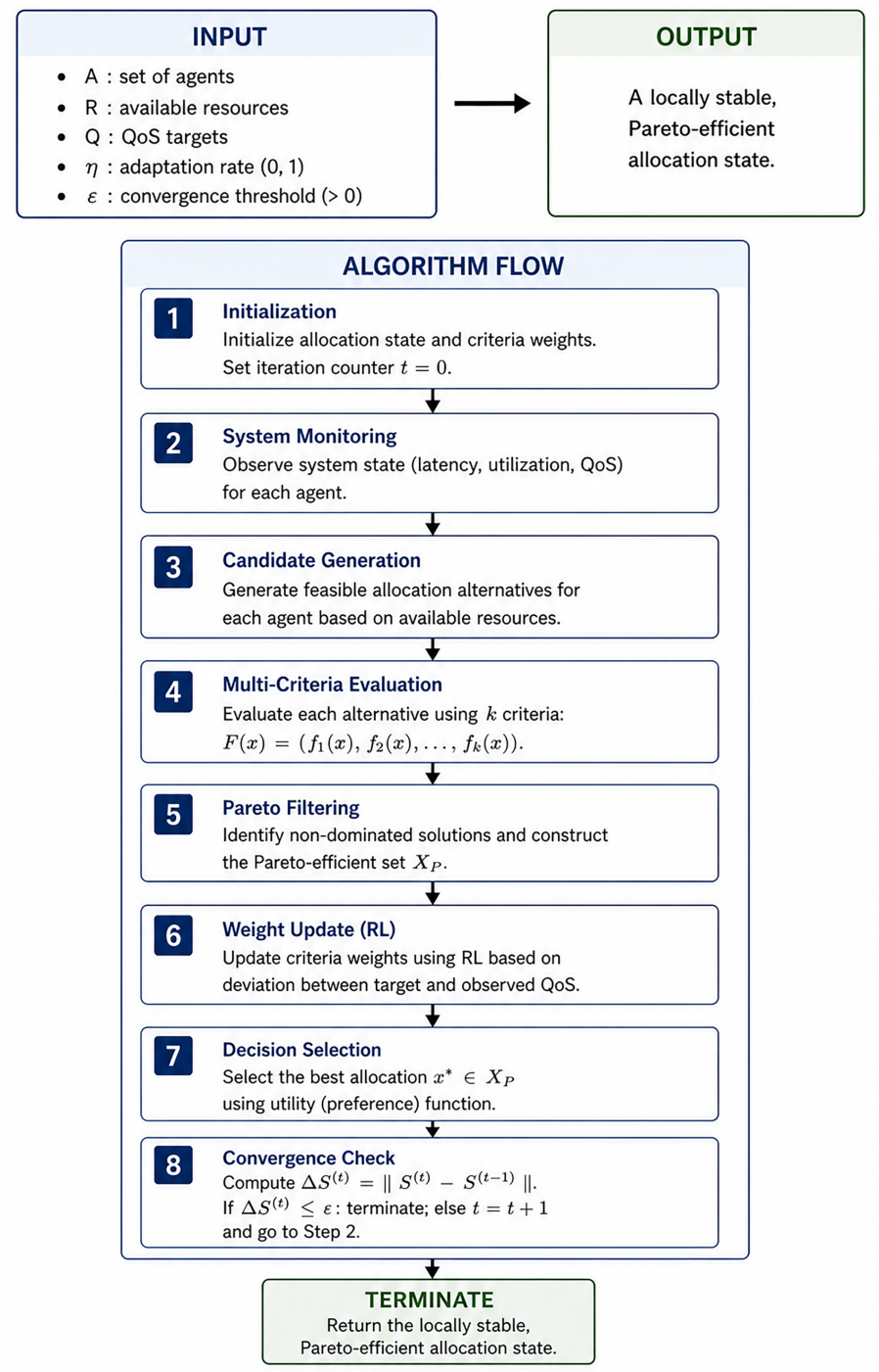


Figure 3: Workflow of the adaptive Pareto-based multi-agent resource allocation algorithm, illustrating system state monitoring, candidate allocation generation, Pareto-based evaluation, QoS-driven adaptive priority updating, decentralized agent decision-making, and iterative convergence.

In Algorithm 1, $S(t)$ denotes the allocation state at iteration t, and $W(t)$ represents the vector of decision criteria weights. Each agent generates candidate allocations and selects a Pareto-efficient solution based

on multi-criteria evaluation. The algorithm iteratively updates the global state and terminates when the convergence condition $\Delta(t) < \varepsilon$ is satisfied.

Algorithm 1: Adaptive Pareto-based agent placement algorithm

Start Algorithm.
1. Input A = {a1, ..., an}, R = {r1, ..., rm}, Q*, α, ε.
2. Initialize allocation state S(0) and weights W(0).
3. Set iteration counter t = 0.
4. repeat
5. for each agent ai ∈ A do
6. Observe system state: (Li(t), Ui(t), Qi(t)).
7. Generate candidate set Ci(t) ⊆ R.
8. Evaluate candidates: F(c) = (f_1(c), f_2(c), ..., f_m(c)).
9. Determine Pareto set: Pi(t) = {c ∈ Ci(t) | ∄c′ ∈ Ci(t): c′ > c}
10. Update weights: wj(t + 1) = wj(t) + α|qj* − qj(t)|
11. Select si(t + 1) ∈ Pi(t).
12. end for
13. Update global state S(t + 1).
14. Compute Δ(t) = ||S(t + 1) − S(t)||.
15. t = t + 1.
16. until Δ(t) < ε
17. Output S*.
End Algorithm.

The proposed algorithm relies on local decision-making by agents without centralized control, enabling stable and adaptive resource utilization in real-time BI environments.

4.2 Computational Complexity and Scalability Analysis

The proposed algorithm operates independently on each agent without centralized coordination, with computational complexity depending on the number of agents and candidate placement options per iteration. At each step, agents generate alternative placements based on available resources and evaluate them using Pareto dominance. Thus, if the system consists of agents, the computational complexity of a single iteration can be estimated as:

$$O\left(n \cdot m^2\right) \tag{8}$$

where n denotes the total number of agents, and m denotes the number of alternative placement options evaluated by each agent.

This complexity estimate also reflects the computational overhead of the decentralized coordination process, which remains manageable due to the absence of centralized global optimization procedures.

5 Experimental Evaluation

This section evaluates the performance of the Pareto-based agent placement algorithm in cloud BI environments, focusing on its effectiveness in improving resource utilization and reducing response latency for BI services.

5.1 Experimental Setup

The experimental evaluation was conducted using a discrete-event cloud simulation environment. The simulation framework was developed based on principles similar to the CloudSim architecture, allowing flexible configuration of processing capacity, memory, and network resources for virtual computing nodes. Such simulation-based approaches are widely adopted in cloud computing research to evaluate resource management and optimization algorithms under controlled, reproducible, and scalable conditions. The detailed experimental parameters and configurations used in this study are summarized in Table 2.

Table 2: Experimental parameters and settings.

Parameter	Value
Number of agents	10–50
Number of cloud nodes	5–20
CPU capacity per node	4–8 vCPU
Memory per node	8–32 GB
Workload type	Dynamic BI queries (variable arrival rate)
Baseline methods	Static, Random, GA (default settings)

During the experiments, the following key performance metrics were monitored:

- average response latency of BI services.
- utilization level of computational and memory resources.
- stability of service quality.

Each simulation scenario was executed ten independent times under identical parameter settings, and the reported metrics correspond to average values across repeated runs, ensuring fair and consistent evaluation conditions.

The workload generation process emulated dynamically changing BI query streams, including concurrent analytical requests, variable service interaction intensity, and fluctuating resource utilization patterns commonly observed in cloud-based BI environments.

All simulation scenarios were executed under identical resource availability assumptions and decentralized agent coordination settings to ensure reproducibility and consistency across baseline comparisons.

5.1.1 Real Cloud Testbed Setup

A small-scale Kubernetes-based testbed was deployed to complement simulation results, with its configuration summarized in Table 3. Service response time and CPU utilization were selected as the primary performance metrics and compared with simulation-based results to analyze the algorithm's adaptability under real-world conditions.

These baselines complement the simulation-based comparison and ensure consistency across experimental scenarios.

The Kubernetes deployment is implemented as a pilot single-node setup due to resource constraints. However, the proposed algorithm is inherently decentralized and designed for scalable multi-node cloud environments. To partially validate this claim within the scope of the current study, the simulation experiments were designed to reflect multi-node scenarios with up to 40 cloud nodes and 100 agents operating

concurrently, and the results demonstrate stable performance consistent with the theoretical scalability analysis in Section 4.2.

Table 3: Real cloud testbed configuration.

Component	Description
Platform	Kubernetes (Minikube)
Number of nodes	1 (single-node testbed)
Number of pods	3–5
BI services	Deployed as service pods
Agent controller	Agent coordination and monitoring pod
Metrics	Response time (ms), CPU utilization (%)

5.1.2 Reproducibility Settings

The workload generation process was implemented using dynamically varying BI analytical query streams with periodic workload bursts to emulate real-time cloud-based BI environments. Query arrival intensity varied between 20 and 100 requests per minute during simulation runs. The workload generation model included concurrent analytical requests and dynamically changing resource utilization patterns.

Unless otherwise specified, the adaptive learning rate and convergence threshold were fixed at $\alpha = 0.3$ and $\varepsilon = 0.01$, respectively. Each agent generated between 5 and 10 candidate allocation alternatives during every iteration step. The maximum number of iterations for each simulation scenario was limited to 100 iterations.

All experiments were executed under identical resource availability conditions and repeated 10 independent times using fixed random seeds to ensure reproducibility and consistency of results. The simulation environment assumed stable network conditions and homogeneous cloud nodes without node failure simulation.

Table 4 summarizes the main reproducibility-related parameter settings used throughout the experimental evaluation.

Table 4: Reproducibility and simulation parameter settings.

Parameter	Value
Adaptation rate α	0.3
Convergence threshold ε	0.01
Request arrival rate	20–100 requests/min
Candidate allocations per agent	5–10
Maximum iterations	100
Simulation repetitions	10
Random seed policy	Fixed seeds
Network conditions	Stable
Cloud node configuration	Homogeneous

These settings were kept consistent across all experimental scenarios to ensure fair comparison and reproducibility of the obtained results.

5.2 Evaluation Metrics and Baseline Methods

The evaluation metrics include decision accuracy, QoS violation rate, resource utilization efficiency, and convergence time. These metrics collectively reflect SLA compliance, system throughput, and overall system stability.

Fairness among agents is indirectly supported through Pareto-consistent decentralized allocation decisions that prevent persistent resource monopolization and encourage balanced workload distribution. Computational overhead remains manageable due to the decentralized coordination mechanism, whose iterative complexity is bounded by $O(n \cdot m^2)$as discussed in Section 4.2.

In the context of this study, throughput is reflected by the system's ability to maintain stable response latency and resource utilization under increasing analytical workload intensity.

To evaluate the proposed Pareto-based algorithm, it was compared against four baseline methods: static allocation, random allocation, load-based heuristic, and evolutionary optimization. In addition, a non-adaptive weighted decision model was included as a baseline, in which fixed criterion weights are assigned prior to execution and remain unchanged throughout the allocation process. This baseline enables direct assessment of the benefit introduced by the adaptive weight update mechanism. Static allocation fixes resources in advance without adapting to workload changes, serving as a lower-bound reference. These lightweight baselines are commonly used in cloud BI systems and enable clear assessment of the proposed approach's advantages in real-time multi-objective environments.

5.2.1 Random Allocation

In the random allocation approach, available cloud resources are assigned to agents without any optimization strategy. This method serves as a baseline to demonstrate the minimum performance achievable in the absence of structured decision-making.

5.2.2 Genetic Algorithm

To compare the proposed Pareto-based agent placement algorithm with a more powerful optimization technique, an evolutionary optimization-based Genetic Algorithm (GA) was employed as a baseline method. A Genetic Algorithm (GA) was employed as a baseline method, with parameters summarized in Table 5.

Table 5: Genetic algorithm parameters used in the baseline evaluation.

Parameter	Value
Population size	30
Number of generations	50
Selection method	Tournament selection
Crossover rate	0.8
Mutation rate	0.1
Elitism	Enabled

Fig. 4 illustrates the comparative performance of the proposed approach against baseline methods, with quantitative results summarized in Table 6.

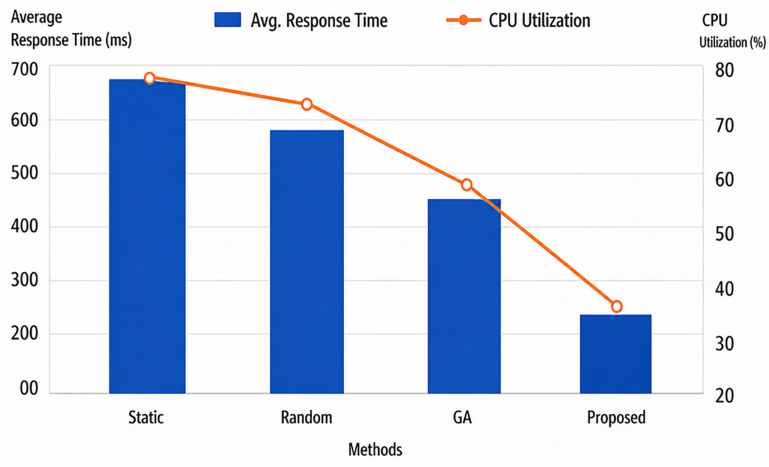


Figure 4: Performance comparison of the Pareto-based agent placement approach with baseline methods.

Table 6: Comparison of response latency and resource utilization across agent placement strategies.

Method	Mean Latency (ms)	Latency Std (ms)	95% CI	CPU Utilization (%)	CPU Std (%)
Static	185	9.4	[178.3, 191.7]	78	4.2
Random	132	7.1	[126.9, 137.1]	65	3.8
GA	108	6.2	[103.6, 112.4]	57	3.1
Pareto	84	4.8	[80.6, 87.4]	46	2.4

5.3 Results

All experiments were repeated 10 independent times under identical parameter settings. The reported results correspond to averaged values, and variability is quantified using 95% confidence intervals.

Experimental results confirm that the proposed Pareto-based approach outperforms all baseline methods. Service latency remains consistently lower and more stable under increasing workload, owing to agents' multi-objective Pareto-efficient decisions that balance latency and resource utilization while reducing fluctuations under dynamic workloads.

Fig. 4 presents a comparative analysis of service latency and resource consumption across the proposed and baseline methods. Variability across 10 independent experimental runs is quantified using 95% confidence intervals, as reported in Table 6. The Pareto-based approach ensures balanced resource utilization and improved system performance for real-time BI services. Under dynamic workloads, static allocation leads to sharp latency increases, whereas the Pareto-based approach significantly limits such fluctuations, achieving more stable performance for real-time analytical tasks. Quantitative results are summarized in Table 6.

The proposed method consistently achieves lower QoS violation rates and faster convergence compared to baseline approaches, demonstrating improved SLA compliance and overall system performance.

Statistical comparison between the Pareto-based approach and the GA baseline indicates a statistically significant reduction in response latency (t = 8.37, df = 58, p = 0.0004). The 95% confidence interval of the mean difference was estimated as [18.2, 29.7 ms], which does not include zero, confirming the robustness of the observed improvement. The calculated effect size (Cohen's d = 1.54) indicates a large practical effect according to standard statistical interpretation guidelines. These results demonstrate not only statistical significance but also substantial practical performance gains.

To provide broader robustness validation, repeated-run statistics were additionally analyzed across all baseline methods under identical workload generation and execution settings. The observed variability remained consistent across Static, Random, GA-based, and Pareto-based allocation strategies, while the proposed Pareto-based method maintained the lowest mean latency and standard deviation. Confidence interval analysis across all compared methods further confirmed the stability and robustness of the proposed decentralized coordination mechanism under dynamically changing BI workloads.

To improve robustness, all baseline evaluations were repeated under identical workload generation and execution settings. Consistent performance trends were observed across repeated runs, indicating stable behavior of the proposed decentralized coordination mechanism.

QoS violation rates were observed to decrease consistently under the proposed method, indicating improved SLA compliance.

The results reported in Table 7 indicate that, even under a controlled Kubernetes-based cloud setup, the Pareto-based agent placement approach consistently outperforms static and GA-based baseline methods in terms of service response time and CPU utilization efficiency. These observations are consistent with the simulation-based findings and support the stability-oriented behavior predicted by the theoretical analysis in Section 3.3.

Table 7: Performance comparison of placement strategies in a Kubernetes-based testbed.

Method	Average Response Time (ms)	CPU Utilization (%)
Static Allocation	820	68
GA-based Allocation	690	56
Pareto-based Agent Placement	540	42

5.4 Sensitivity Analysis

To evaluate the scalability of the proposed approach, the number of active agents gradually increased from 10 to 50 while monitoring changes in service response latency and CPU utilization. The results are illustrated in Fig. 5.

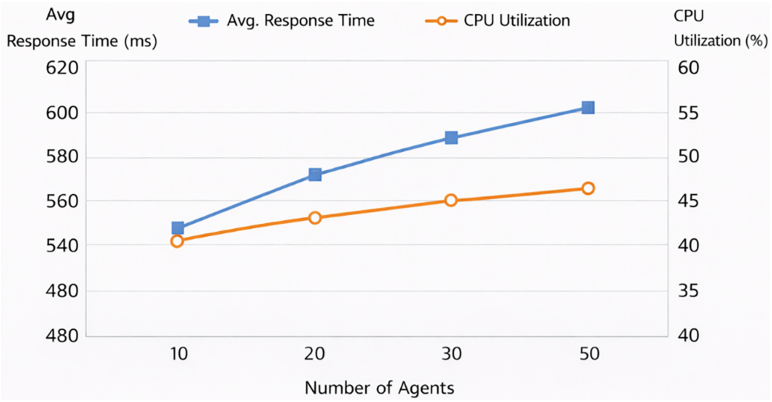
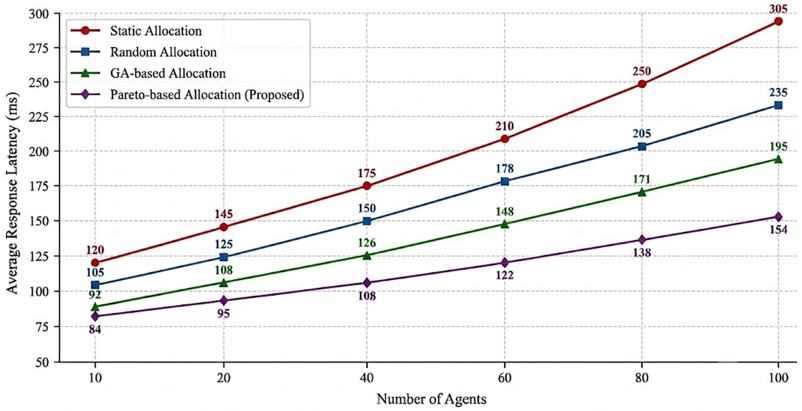


Figure 5: Scalability-oriented sensitivity analysis illustrating response latency and CPU utilization behavior under increasing numbers of agents and cloud nodes.

As shown in Fig. 5, response latency and CPU utilization both increase gradually with the growing number of agents, without abrupt performance degradation, indicating that the Pareto-based approach maintains balanced resource allocation under scaling conditions. Additional experiments were conducted for adaptation rates $\alpha \in \{0.1, 0.3, 0.5, 0.7\}$. Smaller α values resulted in slower adaptation but higher stability, while larger α values accelerated convergence at the cost of minor oscillatory behavior. The most balanced performance was observed for α between 0.3 and 0.5.

To further investigate scalability behavior under larger distributed BI workloads, additional comparative simulations were conducted with progressively increasing numbers of agents. The scalability-oriented comparison presented in Fig. 6 illustrates the response latency behavior of different allocation strategies under increasing workload intensity.



Note: Lower values indicate better scalability (i.e., lower response latency under larger workload and more agents).

Figure 6: Scalability comparison of allocation strategies under increasing numbers of agents in cloud-based BI environments.

As illustrated in Fig. 6, the proposed Pareto-based decentralized coordination mechanism demonstrates more stable scalability behavior compared to baseline methods under progressively increasing agent populations. Although response latency increases across all allocation strategies, the proposed approach maintains the slowest growth trend with more balanced resource utilization, while Static and Random strategies exhibit substantially faster latency growth. The observed sub-linear latency growth confirms that decentralized Pareto-consistent coordination enables adaptive workload distribution without significant degradation of system stability. These results provide empirical support for the framework's scalability characteristics and are consistent with the computational complexity analysis presented in Section 4.2.

6 Discussion

Experimental results show that the Pareto-based agent deployment approach has the potential to significantly improve resource management efficiency in cloud-based BI environments. The reduction in service response latency is explained by the ability of agents to simultaneously consider multiple conflicting criteria in their decision-making process. In contrast, static and random resource allocation strategies lack such flexibility, which limits the ability of the system to respond effectively to changes in workload. Pareto-efficient solutions selected by agents reduce resource overload and help maintain a relatively balanced operation mode in the cloud infrastructure.

Fairness among agents is indirectly supported through Pareto-consistent allocation decisions, which prevent persistent resource monopolization and promote balanced workload distribution across competing services.

The proposed multi-agent resource management model allows BI systems to be developed as flexible and scalable platforms. The independent operation of agents within a microservices-based architecture simplifies the integration of new analytical services and the dynamic reconfiguration of existing services.

Unlike RL-based schedulers that depend on continuous policy optimization and global state synchronization, the proposed approach enables decentralized local adaptation based on QoS deviations. This reduces coordination overhead while maintaining stable multi-criteria resource allocation under dynamic BI workloads.

The experimental part of this study was carried out in a Kubernetes environment deployed via Minikube. However, we acknowledge that this configuration represents a single-node cluster and does not replicate the full behavior of a distributed production cloud. Therefore, the results presented reflect a controlled experimental investigation rather than a large-scale production deployment. Further experiments on multi-node Kubernetes clusters with heterogeneous node configurations are warranted in the future.

To assess the scalability behavior of the proposed algorithm under larger configurations, additional simulation experiments were conducted with agent counts ranging from 10 to 100 and cloud node counts from 5 to 40. The results indicate that the average response latency increases sub-linearly with the number of agents, and CPU utilization remains within acceptable bounds across all tested configurations. These findings are consistent with the $O(n \cdot m^2)$ complexity estimate presented in Section 4.2 and suggest that the algorithm scales gracefully within the simulated environment. Evaluation in a large-scale production deployment (e.g., AWS or Azure) remains a direction for future work.

Although full-scale production cloud experiments were beyond the scope of the current study, the scalability-oriented simulations with increasing numbers of agents and cloud nodes demonstrated stable performance behavior under progressively larger configurations. Although the real-cloud validation was conducted in a controlled single-node Kubernetes environment, the proposed decentralized coordination mechanism does not depend on centralized orchestration and is inherently designed for distributed multi-node scalability. Therefore, the additional large-scale simulation experiments provide partial empirical support for the scalability properties predicted by the theoretical complexity analysis.

Threats to Validity

Several threats to the validity of this study should be acknowledged. Regarding internal validity, the simulation environment was implemented as a custom discrete-event framework, which may not capture all real-world scheduling behaviors. Regarding external validity, the real cloud validation was conducted on a single-node Minikube setup, which does not replicate the full behavior of a distributed production cloud environment. Multi-node Kubernetes clusters with heterogeneous configurations represent an important direction for future validation. Nevertheless, the scalability-oriented simulation experiments presented in Section 5.4 partially mitigate this limitation by evaluating the proposed decentralized coordination mechanism under progressively increasing agent populations and cloud node configuration. Regarding construct validity, the selected performance metrics (latency, CPU utilization, QoS violation rate, convergence time) reflect the primary objectives of the study; however, additional metrics such as energy consumption and cost efficiency were not evaluated due to infrastructure limitations.

7 Conclusions

This paper addressed the problem of resource management and service placement in cloud-based BI systems and proposed a decentralized Pareto-based multi-agent approach to tackle this challenge. The proposed model is specifically designed to improve resource utilization efficiency under real-time analytical workloads and microservices-based architectures that are characteristic of BI systems.

The results demonstrate that local Pareto-efficient decisions made by agents ensure balanced resource allocation in cloud environments and significantly reduce service response latency. By avoiding reliance on centralized control mechanisms, the proposed approach offers important scientific and practical benefits in terms of scalability and stability of BI systems. The obtained results further demonstrate that decentralized Pareto-consistent adaptive coordination can provide a computationally lightweight and scalable alternative to centralized or training-intensive cloud resource allocation approaches in dynamic BI microservice environments. Future research will focus on evaluating the proposed model in large-scale production cloud environments such as AWS or Azure, and on extending the optimization framework to incorporate additional objectives including energy efficiency, operational cost, and fault tolerance.

Acknowledgement: The authors are grateful for the scholarly atmosphere and institutional resources made available through Bukhara State University and Gachon University, which supported the completion of this work. We confirm that AI tools were not used to generate, fabricate, or manipulate any scientific data, results, or intellectual content. All research and conclusions are solely the authors' work. AI was used only for improving figure clarity and language editing, with no effect on the scientific integrity of the manuscript.

Funding Statement: The authors received no specific funding for this study. The article processing charge (APC) was funded by the corresponding author.

Author Contributions: Contributions to the paper are as follows: research idea and design—Khamza Eshankulov, Rashid Nasimov; methodology and formal analysis—Khamza Eshankulov, Bahodir Muminov, Zavqiddin Temirov; data collection and experimental evaluation—Robiya Farmonova, Dilnavoz Sodikova, Bakhriddin Bozorov; analysis and interpretation of results—Khamza Eshankulov, Bahodir Muminov, Rashid Nasimov; preparation of the draft paper—Khamza Eshankulov, Zavqiddin Temirov, Rashid Nasimov. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data and implementation code that support the findings of this study will be made publicly available upon acceptance. Materials are available from the corresponding author, Rashid Nasimov, upon reasonable request prior to publication. The implementation code and experimental scripts used in this study will be made publicly available upon acceptance of the manuscript. The repository will include the simulation environment, agent placement algorithm, baseline implementations, and scripts for reproducing all reported figures and tables. Raw experimental data will also be provided. In the meantime, all materials are available from the corresponding author upon reasonable request.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chen H, Chiang RHL, Storey VC. Business intelligence and analytics: from big data to big impact. MIS Q. 2012;36(4):1165–88. doi:10.2307/41703503.
2. Jiménez-Partearroyo M, Medina-López A. Leveraging business intelligence systems for enhanced corporate competitiveness: strategy and evolution. Systems. 2024;12(3):94. doi:10.3390/systems12030094.

3. Hashem IAT, Yaqoob I, Anuar NB, Mokhtar S, Gani A, Ullah Khan S. The rise of "big data" on cloud computing: review and open research issues. Inf Syst. 2015;47(2):98–115. doi:10.1016/j.is.2014.07.006.
4. Dorri A, Kanhere SS, Jurdak R. Multi-agent systems: a survey. IEEE Access. 2018;6:28573–93. doi:10.1109/ACCESS.2018.2831228.
5. Ma N, Tang A, Xiong Z, Jiang F. A deep multi-agent reinforcement learning approach for the micro-service migration problem with affinity in the cloud. Expert Syst Appl. 2025;273(8):126856. doi:10.1016/j.eswa.2025.126856.
6. Akramov TA, Israilov TM, Burkhanov AU, Potapov FP. The mechanism of automatization of smart business through decision-making and management of production based on big data and AI. In: Big data and artificial intelligence for decision-making in the smart economy. Cham, Switzerland: Springer Nature; 2025. p. 33–42. doi:10.1007/978-3-031-78686-0_4.
7. Abdullahi M, Ngadi MA, Abdulhamid SM. Symbiotic organism search optimization based task scheduling in cloud computing environment. Future Gener Comput Syst. 2016;56(3):640–50. doi:10.1016/j.future.2015.08.006.
8. Afrin M, Jin J, Rahman A, Tian YC, Kulkarni A. Multi-objective resource allocation for edge cloud based robotic workflow in smart factory. Future Gener Comput Syst. 2019;97(2):119–30. doi:10.1016/j.future.2019.02.062.
9. Kareem Awad W, Zainol Ariffin KA, Ahmad Nazri MZ, Yassen ET. Resource allocation strategies and task scheduling algorithms for cloud computing: a systematic literature review. J Intell Syst. 2025;34(1):20240441. doi:10.1515/jisys-2024-0441.
10. Pérez M, Cañizares PC, Núñez A. Multi-objective optimization of cloud systems. Sci Comput Program. 2026;252(1):103447. doi:10.1016/j.scico.2026.103447.
11. Mishra S, Sangaiah AK, Sahoo MN, Bakshi S. Pareto-optimal cost optimization for large scale cloud systems using joint allocation of resources. J Ambient Intell Humaniz Comput. 2023;14(11):15375–93. doi:10.1007/s12652-019-01601-x.
12. Rani R, Garg R. Pareto based ant lion optimizer for energy efficient scheduling in cloud environment. Appl Soft Comput. 2021;113:107943. doi:10.1016/j.asoc.2021.107943.
13. Beloglazov A, Buyya R. Energy efficient resource management in virtualized cloud data centers. In: Proceedings of the 2010 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing; 2010 May 17–20; Melbourne, Australia. p. 826–31. doi:10.1109/ccgrid.2010.46.
14. Eshankulov K, Shafiev T, Bakaev I, Nazarov S. Architecture and data structures of the information monitoring and decision-making software suite for business intelligence. In: Proceedings of SPIE Conference on Fourth International Conference on Optics, Computer Applications, and Materials Science (CMSD-IV 2024). Bellingham, WA, USA: SPIE; 2025. p. 39. doi:10.1117/12.3061943.
15. Selvarajan GP. Transforming business intelligence in the cloud era: leveraging scalability and real-time analytics for enhanced decision-making. Int J Enhanc Res Manag Comput Appl. 2023:53–63.
16. Rakhimov M, Ochilov M, Javliev S, Nasimov R. Analysis and possibilities of parallel approach in big data processing. In: Proceedings of the 8th International Conference on Future Networks & Distributed Systems; 2024 Dec 11–12; Marrakech, Morocco. New York, NY, USA: ACM; 2024. p. 20–5. doi:10.1145/3726122.3726126.
17. Nawrocki P, Grzywacz M, Sniezynski B. Adaptive resource planning for cloud-based services using machine learning. J Parallel Distrib Comput. 2021;152(6):88–97. doi:10.1016/j.jpdc.2021.02.018.
18. Shrimali B, Patel H. Multi-objective optimization oriented policy for performance and energy efficient resource allocation in cloud environment. J King Saud Univ Comput Inf Sci. 2020;32(7):860–9. doi:10.1016/j.jksuci.2017.12.001.
19. Nuthakki P, Pavan Kumar T, Alhussein M, Aurangzeb K, Anwar MS, Gunnam LC. AI-driven resource and communication-aware virtual machine placement using multi-objective swarm optimization for enhanced efficiency in cloud-based smart manufacturing. Comput Mater Contin. 2024;81(3):4743–56. doi:10.32604/cmc.2024.058266.
20. Rathee A, Dalal S. A systematic literature review of machine learning-based resource allocation techniques in cloud computing. Computing. 2025;107(9):179. doi:10.1007/s00607-025-01526-8.
21. Bernard L, Yassa S, Alouache L, Romain O. Efficient Pareto based approach for IoT task offloading on fog–cloud environments. Internet Things. 2024;27(4):101311. doi:10.1016/j.iot.2024.101311.

22. Gil Herrera J, Botero JF. Resource allocation in NFV: a comprehensive survey. IEEE Trans Netw Serv Manag. 2016;13(3):518–32. doi:10.1109/TNSM.2016.2598420.
23. Schardong F, Nunes I, Schaeffer-Filho A. NFV resource allocation: a systematic review and taxonomy of VNF forwarding graph embedding. Comput Netw. 2021;185(3):107726. doi:10.1016/j.comnet.2020.107726.
24. Bo Y, Feng J, Zhang S, Leng B. Multi-agent reinforcement learning with graph representation for green edge–cloud computation offloading. Comput Commun. 2025;239(1):108176. doi:10.1016/j.comcom.2025.108176.
25. Pietri I, Sakellariou R. A Pareto-based approach for CPU provisioning of scientific workflows on clouds. Future Gener Comput Syst. 2019;94:479–87. doi:10.1016/j.future.2018.12.004.
26. Eshankulov K, Zohirov K, Bakaev I, Tursun S, Shakhzod N, Temirov Z, et al. An RL-enhanced multi-agent framework for scalable and intelligent business intelligence systems. Information. 2026;17(3):252. doi:10.3390/info17030252.
27. Kitchenham B. Guidelines for performing systematic literature reviews in software engineering. Staffordshire, UK: Keele University; 2007. Report No.: EBSE-2007-01.