An Intelligent System for Energy-Aware Flexible Fault Tolerance in Cloud

Moin Hasan[1], Shailee Lohmor Choudhary[2], Rinku Sharma Dixit[3], Sujith Jayaprakash[4]
Dept. of Data Science & AI, Royal Roads University (RAK Campus), Ras Al Khaimah, UAE[1]
Dept. of Artificial Intelligence and Machine Learning, New Delhi Institute of Management, New Delhi, India[2, 3]
Dept. of ICT, Britts Imperial University College, Sharjah, UAE[4]

***Abstract*—Fault tolerance in cloud computing environments has matured significantly, and most cloud service providers now offer highly available and reliable services. However, such high levels of fault tolerance are often achieved through resource replication, which consequently increases overall energy consumption. Existing research primarily relies on conventional energy management techniques when designing fault-tolerant cloud systems. With recent advancements in artificial intelligence, there is considerable potential to improve energy management beyond traditional approaches. Motivated by this opportunity, this study presents an intelligent system for energy-aware and flexible fault tolerance in cloud environments. The proposed system employs an intelligent resource group formation algorithm that selects virtual machines based on both their energy consumption and reliability characteristics. Furthermore, by dynamically selecting appropriate fault tolerance mechanisms, the system adapts to changing workload conditions and system states. The proposed approach is evaluated through extensive simulation experiments. The results indicate that the system completes more than 96% of tasks within their specified deadlines while maintaining energy-efficient operation. Consequently, the proposed solution effectively balances the trade-off between energy efficiency and service reliability, making it a practical and sustainable approach for modern cloud computing environments.**

Keywords*—Cloud computing; energy-aware computing; fault tolerance; artificial intelligence; service reliability; sustainable computing

I. Introduction

Cloud computing has transformed the way organizations deploy, manage, and scale computing resources. By providing on-demand access to virtualized infrastructure, platforms, and software services, cloud environments enable organizations to reduce capital expenditure while improving operational flexibility and scalability [1]. The widespread adoption of cloud computing across industries has led to the rapid growth of large-scale data centers that support a diverse range of applications and services.

Despite its advantages, cloud computing faces significant challenges related to service reliability and energy consumption. Failures are inevitable in large-scale cloud infrastructures due to hardware malfunctions, software defects, network disruptions, and power-related issues [2]. Such failures can negatively affect application performance, violate service-level agreements (SLAs), and reduce user satisfaction. Consequently, fault tolerance has become a critical requirement for modern cloud systems.

Various fault tolerance mechanisms have been proposed in the literature to address these challenges, including replication, checkpointing, migration, and their hybrid variants [3]. Replication enhances reliability by maintaining multiple copies of tasks or services, while checkpointing periodically saves application states to enable recovery after failures. Migration techniques proactively move workloads away from unreliable resources to minimize service disruption. Although these approaches improve system reliability, they often incur substantial resource and energy overheads [3]. In particular, replication-based solutions may significantly increase energy consumption due to the simultaneous execution of redundant resources.

At the same time, energy efficiency has emerged as an equally important concern in cloud computing. Data centers consume an enormous amount of electrical power to operate computing, storage, networking, and cooling infrastructure [4]. This growing energy demand not only increases operational costs for cloud providers but also contributes to environmental concerns associated with carbon emissions [5]. Traditional energy management techniques such as virtual machine (VM) consolidation [6] and dynamic voltage and frequency scaling (DVFS) [7] have shown promise in reducing energy consumption; however, they are often applied independently of fault tolerance strategies.

Most of the existing studies focus primarily on either improving reliability or reducing energy consumption. As a result, the interdependence between these two objectives is frequently overlooked. Fault tolerance mechanisms designed without considering energy implications may lead to excessive power consumption, while aggressive energy-saving strategies may compromise system reliability. Furthermore, static fault tolerance approaches lack the flexibility required to adapt to dynamic workload characteristics and changing cloud conditions [8].

To address these limitations, this study proposes an intelligent system for energy-aware flexible fault tolerance in cloud computing environments. The proposed approach incorporates an intelligent resource group formation algorithm that evaluates VMs based on both reliability and energy-related characteristics. By dynamically selecting appropriate fault tolerance mechanisms according to task requirements and system conditions, the system aims to achieve a balanced

trade-off between service reliability and energy efficiency. The novelty of the proposed work does not lie in introducing a completely new replication or checkpointing mechanism. Rather, the contribution is an integrated decision framework that simultaneously 1) formulates energy-aware fault tolerance mathematically, 2) introduces energy-aware resource group formation, and 3) dynamically selects fault tolerance mechanisms according to workload characteristics, reliability requirements, and system state.

The proposed system is evaluated through extensive simulation experiments under diverse workload scenarios. Initial results demonstrate that the system successfully maintains a high task completion rate while controlling energy consumption, thereby providing an effective and sustainable solution for fault-tolerant cloud computing.

The remainder of this study is organized as follows. Section II reviews related work on cloud fault tolerance and energy-aware resource management. Section III presents the flexible fault tolerance model, followed by the energy-aware resource group formation algorithm in Section IV. Section V describes the proposed system, explaining its component-wise architecture. Results and discussion are covered in Section VI. Section VII concludes the study with directions for future research.

II. Related Works

Fault tolerance has been extensively studied in cloud computing due to the increasing demand for highly available and reliable services. Traditional fault tolerance mechanisms such as replication, checkpointing, task resubmission, and VM migration have been widely adopted to mitigate the impact of hardware failures, software faults, and network disruptions. Recent research has focused on improving the adaptability and efficiency of these mechanisms through intelligent scheduling, predictive analytics, and optimization-based resource management.

Several studies have explored proactive fault tolerance approaches that predict failures before they occur. Patil and Patil [9] proposed a dynamic load balancing model capable of detecting workload spikes and preventing host overloads through proactive migration strategies. Similarly, Ray et al. [10] introduced a fault management framework for cloud federation environments that predicts failures using CPU temperature monitoring and proactively migrates VMs to healthy cloud service providers. More recently, Ragmani et al. [11] employed artificial neural networks to predict hard disk failures, enabling preventive VM migration and improving service availability. Log-based anomaly detection techniques have also been investigated, where machine learning and large language models are utilized to identify failure indicators from system logs and initiate fault recovery actions before service degradation occurs [12].

Another stream of research focuses on adaptive and flexible fault tolerance. Kumari and Kaur [13] developed a fuzzy logic-based framework that dynamically selects between replication and checkpointing according to machine failure risk, storage availability, and task priority. Likewise, Mushtaq et al. [14] proposed an integrated scheduling framework that combines replication and checkpointing while dynamically adapting to VM-level and task-level failures. These approaches demonstrate that adaptive fault tolerance can significantly improve resource utilization and recovery performance compared with static mechanisms. However, their decision-making processes are primarily guided by reliability and performance considerations, with limited attention to energy implications.

Resource-aware fault tolerance has also received considerable attention. Dehury et al. [15] proposed a rank-based fault tolerance strategy that allocates replicas according to component criticality, thereby reducing unnecessary resource consumption. Similarly, cluster-centric scheduling frameworks and reservation-based fault tolerance strategies have been developed to improve Quality of Service (QoS), reliability, and task completion rates [16], [17]. While these methods reduce resource overhead compared with conventional replication schemes, energy consumption is rarely incorporated as a primary optimization objective.

In parallel, substantial research efforts have focused on energy-efficient cloud computing. Traditional approaches such as DVFS, VM consolidation, and intelligent workload scheduling have demonstrated significant reductions in power consumption. More recently, artificial intelligence and optimization techniques have been employed to further enhance energy management. Evolutionary programming, reinforcement learning, deep learning, and bio-inspired optimization algorithms have been successfully applied to dynamic resource allocation and workload scheduling [18], [19], [20], [21], [22]. For example, reinforcement learning-based frameworks have achieved considerable reductions in energy consumption and carbon emissions through intelligent power management [20], while hybrid machine learning models have improved energy-efficient task scheduling in cloud and fog environments [22].

Despite these advances, existing energy-aware solutions primarily focus on resource optimization, workload placement, or carbon reduction, assuming stable cloud infrastructures [18], [19], [20]. Fault tolerance is often treated as an independent concern. Conversely, fault-tolerant systems frequently prioritize reliability, availability, and SLA compliance without explicitly considering their energy overhead [13], [14], [15], [16]. Replication-based techniques, in particular, can substantially increase energy consumption due to redundant resource utilization, while migration and checkpointing mechanisms introduce additional computational and communication costs.

A review of the literature reveals several important research gaps. First, there is a lack of integrated frameworks that simultaneously address fault tolerance and energy efficiency. Second, although adaptive fault tolerance mechanisms exist, their decision-making processes rarely consider energy consumption as a key factor [13], [14]. Third, existing multi-objective optimization models generally focus on a limited set of objectives, such as reliability, migration cost, or makespan, without jointly considering energy efficiency, reliability, and deadline requirements [10], [19], [20], [23]. Finally, most studies evaluate performance and reliability independently,

with limited investigation of the trade-off between energy consumption and fault tolerance effectiveness.

To address these limitations, this study proposes an intelligent energy-aware flexible fault tolerance system for cloud computing environments. Unlike existing approaches that treat energy management and fault tolerance separately, the proposed framework integrates both objectives into a unified decision-making process. By considering reliability characteristics, energy consumption, workload requirements, and system conditions simultaneously, the proposed system aims to achieve an effective balance between service reliability and sustainable cloud operation.

III. FLEXIBLE FAULT TOLERANCE MODEL

This Section presents a comprehensive mathematical model of an energy-aware flexible fault tolerance framework for cloud computing environments. The model captures the relationships between physical hosts, VMs, and user tasks while accounting for execution dynamics, failure probabilities, and energy consumption at both VM and host levels. By formally modeling energy costs associated with different fault tolerance mechanisms, the proposed framework enables informed trade-off analysis between reliability improvement and energy efficiency. The resulting model provides a strong analytical foundation for the multi-objective optimization and algorithm design stages introduced in the subsequent Sections.

At the infrastructure level, the cloud data center consists of a finite set of physical hosts [Eq. (1)]. Each physical host represents a physical machine equipped with processing, memory, storage, and networking capabilities [24]. To enable elastic resource provisioning [25], a physical host dynamically supports a set of VMs, as shown in Eq. (2). Virtualization allows efficient sharing of physical resources while providing isolation among workloads [26]. The number and configuration of VMs on a host may vary over time depending on workload demand and system conditions.

$$H = \{h_1, h_2, h_3, \dots, h_m\} \quad (1)$$

$$VM = \{v_1, v_2, v_3, \dots, v_n\} \quad (2)$$

Each VM is characterized by a multidimensional resource vector consisting of CPU capacity (v_j^{cpu}), memory size (v_j^{mem}), available bandwidth (v_j^{bw}), and reliability (v_j^{rel}) as defined in Eq. (3). These resources are allocated to user-submitted tasks and directly influence execution performance, energy consumption, and fault tolerance behavior.

$$v_j = (v_j^{cpu}, v_j^{mem}, v_j^{bw}, v_j^{rel}) \quad (3)$$

Users submit a set of tasks or cloudlets for execution, represented in Eq. (4). These tasks may correspond to compute-intensive, data-intensive, or mixed workloads and are executed on the available VMs in the cloud data center [27]. Since cloud environments are inherently dynamic and failure-prone, each task requires fault tolerance support. Rather than relying on a single rigid fault tolerance technique, the system adopts flexible fault tolerance mechanisms, such as task replication, checkpointing, and VM migration. The choice of mechanism depends on runtime system conditions, failure risk, and energy considerations.

$$T = \{t_1, t_2, t_3, \dots, t_p\} \quad (4)$$

Each task t_k in the task set is formally described using a set of parameters, including arrival time (t_k^{at}), number of instructions (t_k^{ni}), deadline (t_k^{dl}), execution start time (t_k^{st}), execution time (t_k^{et}), and expected completion time (t_k^{ct}) as presented in Eq. (5). These parameters enable precise modeling of task scheduling, deadline constraints, and quality-of-service (QoS) requirements.

$$t_k = (t_k^{at}, t_k^{ni}, t_k^{dl}, t_k^{st}, t_k^{et}, t_k^{ct}) \quad (5)$$

When a task t_k is executed on a VM v_j, its execution time depends on the computational capacity of the VM. The execution time of the task is computed using Eq. (6), which relates the number of instructions to the processing speed of the VM. Based on this execution time, the expected completion time of the task is calculated using Eq. (7). These expressions allow the system to determine whether task deadlines can be met under different fault tolerance strategies [28].

$$t_k^{et} = \frac{t_k^{ni}}{v_j^{cpu}} \quad (6)$$

$$t_k^{ct} = t_k^{st} + t_k^{et} \quad (7)$$

Energy consumption is a critical concern in cloud data centers due to rising operational costs and sustainability requirements. For a given physical host h_i, the instantaneous power consumption $h_i^{pow\_co}(t)$ is modeled as a function of its idle power ($h_i^{pow\_id}$), maximum power ($h_i^{pow\_mx}$), and CPU utilization ($h_i^{cpu\_ut}(t)$) at time t [Eq. (8)], provided, $h_i^{cpu\_ut}(t) \in [0,1]$. This widely adopted linear power model captures the strong correlation between CPU utilization and power usage in modern servers [29].

$$h_i^{pow\_co}(t) = h_i^{pow\_id} + (h_i^{pow\_mx} - h_i^{pow\_id}).\, h_i^{cpu\_ut}(t) \quad (8)$$

Using the instantaneous power model, the total energy consumed by a host h_i over the interval $[0, T]$ is computed by integrating power consumption over time, as shown in Eq. (9). This formulation accurately reflects energy usage during task execution, VM migration, and fault tolerance operations. The overall energy consumption of the cloud data center is then obtained by summing the energy consumed by all hosts, as expressed in Eq. (10).

$$h_i^{enr\_co} = \int_0^T h_i^{pow\_c0}(t)\, dt \quad (9)$$

$$H^{enr\_co} = \sum_{i=1}^{m} h_i^{enr\_co} \quad (10)$$

In a cloud environment, VMs are subject to failures caused by hardware degradation, resource contention, overheating, or software anomalies [3]. Each VM v_j configured over the host h_i is therefore associated with a failure probability ($v_j^{pr\_fl}$) which can be estimated using historical failure data, utilization trends, thermal indicators, or log-based anomaly detection techniques. When a task t_k is replicated across multiple (r) independent VMs to improve reliability, the probability that the task completes successfully increases. The resulting task reliability is mathematically defined in Eq. (11), assuming independent VM failures.

$$t_k^{rel\_ex} = 1 - \prod_{j=1}^{r} v_j^{pr\_fl} \quad (11)$$

While fault tolerance improves reliability, it introduces additional energy overhead. The energy cost of providing flexible fault tolerance to a task is calculated in Eq. (12), where $t_k^{enr\_rp}$, $t_k^{enr\_cp}$, and $t_k^{enr\_mg}$ represent energy consumed for replication, checkpointing, and migration of t_k respectively. This equation captures the cumulative energy consumed by different fault tolerance mechanisms, including task replication, checkpointing operations, and VM migration. By explicitly modeling these energy costs, the framework enables informed trade-off decisions between reliability improvement and energy efficiency.

$$t_k^{enr\_ft} = t_k^{enr\_rp} + t_k^{enr\_cp} + t_k^{enr\_mg} \quad (12)$$

The above mathematical model establishes a unified foundation for analyzing energy-aware flexible fault tolerance in cloud computing environments. It integrates workload characteristics, execution dynamics, failure behavior, and energy consumption into a single framework, thereby enabling the design of intelligent fault tolerance strategies that enhance reliability while minimizing energy overhead. This model serves as the basis for subsequent multi-objective optimization and algorithm design in the proposed framework.

IV. Energy-Aware Resource Group Formation Algorithm for Flexible Fault Tolerance

This Section introduces an intelligent mechanism that dynamically selects and groups cloud resources for fault-tolerant task execution while balancing multiple conflicting objectives, namely energy efficiency, reliability, and deadline satisfaction. The proposed algorithm forms a critical component of the framework and directly influences its overall performance and effectiveness.

A. Problem Definition

For each incoming task submitted to the cloud data center, the system must form a resource group consisting of one or more VMs on which the task will be executed using flexible fault tolerance techniques. The selection of this resource group must satisfy the following requirements:

- The task must complete within its specified deadline.
- The overall task reliability must meet or exceed a minimum required threshold.
- The additional energy consumed due to fault tolerance must be minimized.

Let t_k and RG_k denote the task and its corresponding resource group. The challenge lies in determining the optimal composition of RG_k from the available set of VMs such that all constraints are satisfied while optimizing energy efficiency.

B. Decision Variables and Constraints

To formalize the resource group formation problem, a binary decision variable is defined in Eq. (13), using which the following constraints are imposed.

$$x_{jk} = \begin{cases} 1, if\ v_j\ is\ selected\ for\ t_k \\ 0, otherwise \end{cases} \quad (13)$$

- Reliability Constraint: The reliability achieved by executing the task on its resource group must be no less than the minimum required reliability, i.e., $t_k^{rel\_ex} \geq t_k^{rel\_mn}$.
- Deadline Constraint: The expected completion time of the task must not exceed its deadline, i.e., $t_k^{ct} \leq t_k^{dl}$.
- Resource Capacity Constraint: The total resource demand placed on any VM must not exceed its available capacity, i.e., $\sum_{t_k} x_{jk} . t_k^{cpu} \leq v_j^{cpu}$.
- Replication Limit Constraint: To avoid excessive replication and uncontrolled energy usage, the size of the resource group is bounded, i.e., $|RG_k| \leq r_{max}$.

The resource group formation problem is formulated as a multi-objective optimization problem with the following objectives:

- The total energy consumed for executing and protecting tasks is minimized.
- The overall task reliability is maximized by selecting VMs with higher individual reliability.
- The risk of deadline violation is minimized by ensuring timely task completion.

Since these objectives are conflicting, an exact optimal solution is computationally expensive. Therefore, a heuristic-based approach is adopted to efficiently approximate the Pareto-optimal solution. Because a linear power consumption model introduces a strictly monotonic cost curve relative to host CPU utilization, our heuristic operates as an elegant, computationally efficient approximation of a bounded Pareto-frontier by finding the exact point where replication addition ceases once the constraint $t_k^{rel\_ex} \geq t_k^{rel\_mn}$ is satisfied.

C. Proposed Energy-Aware Resource Group Formation Algorithm

The Energy-Aware Resource Group Formation algorithm heuristically solves the multi-objective optimization problem described above. The algorithm operates in four main stages:

- Filtering of candidate VMs based on deadline and resource feasibility
- Computation of energy–reliability composite scores
- Ranking and selection of VMs
- Adaptive formation of the resource group

For each candidate VM, a composite score is computed to balance energy efficiency and reliability [Eq. (14)], where E_{jk} represents the energy cost incurred when task t_k is executed or replicated on VM v_j.

$$v_j^{score} = \alpha . \frac{1}{E_{jk}} + \beta . v_j^{rel} \quad (14)$$

The proposed algorithm for energy-aware group formation is given below:

Algorithm 1: Energy-Aware Resource Group Formation

Input:

Task set $T = \{t_1, t_2, t_3, \dots, t_p\}$,

VM set $VM = \{v_1, v_2, v_3, \dots, v_n\}$,

Weight factors α and β such that $\alpha + \beta = 1$, and

Maximum replication limit r_{max}

Output:

Resource group RG_k for each task t_k

Pseudo code:

EnergyAwareResourceGroupFormation(T, VM)

For each task $t_k \in T$ do

$RG_k \leftarrow \emptyset$

$Candidate\ VMs \leftarrow \emptyset$

For each VM $v_j \in VM$ do

Estimate execution time t_k^{et} of t_k on v_j

Estimate completion time $t_k^{ct} = current_{time} + t_k^{et}$

If $t_k^{ct} \leq t_k^{dl}$ and resources of v_j are sufficient then

Add v_j to Candidate VMs

End

End

For each $v_j \in Candidate\ VMs$ do

Compute energy cost E_{jk} for executing t_k on v_j

Retrieve reliability v_j^{rel}

Compute composite score: $v_j^{score} = \alpha . \frac{1}{E_{ij}} + \beta . v_j^{rel}$

End

Sort $Candidate\ VMs$ in descending order of v_j^{score}

$t_k^{rel\_ex} \leftarrow 0$

$r \leftarrow 0$

For each $v_j \in Candidate\ VMs$ do

If $r = r_{max}$ then

break

End

Add v_j to RG_k

$r \leftarrow r + 1$

Update Current Reliability: $t_k^{rel\_ex} = 1 - \prod_{j=1}^{r} v_j^{fail\_pr}$

for all $v_j \in RG_k$

If $t_k^{rel\_exec} \geq t_k^{rel\_min}$ then

break

End

End

If $t_k^{rel\_exec} \leq t_k^{rel\_min}$ then

Trigger fallback fault tolerance policy

End

End

return $RG_K\ \forall\ t_k \in T$

The algorithm provides several important advantages. It explicitly incorporates energy consumption into fault tolerance decisions and supports flexible adaptation to different workload and system conditions. It also avoids excessive replication by enforcing replication limits. The computational overhead of Algorithm 1 can be broken down into discrete steps for each task $t_k \in T$:

- Filtering Candidate VMs: Evaluating the processing speed constraint across N virtual machines takes $O(N)$ operations.
- Composite Score Computation: Computing v_j^{score} using (14) takes $O(N)$ time.
- Sorting Candidates: Sorting the candidate subset of size $M \leq N$ requires a worst-case time of $O(MlogM)$.
- Group Allocation Loop: Selecting VMs up to the maximum replication limit r_{max} runs in $O(r_{max})$ constant time since $r_{max} \ll N$.

Thus, for an incoming task sequence of size P, the algorithmic time complexity is given by $O(P.NlogN)$. By integrating reliability and energy awareness into the decision-making, the algorithm bridges the gap between theoretical modeling and practical system implementation.

V. PROPOSED SYSTEM

This Section focuses on the design and operation of the proposed Energy-Aware Flexible Fault Tolerance system, which integrates energy efficiency and reliability into a unified cloud fault tolerance solution. It presents the overall architecture of the system and explains each system component and its role in the fault tolerance process. A component-wise architectural diagram is used to illustrate the interaction between system modules, including task submission, resource monitoring, decision-making, fault handling, and execution management.

A. Component-Wise Architecture Description of the System

The proposed system is designed as a modular and layered framework that integrates energy awareness and flexible fault tolerance into cloud task execution. The architecture consists of several interconnected components, each responsible for a specific function in the fault tolerance and decision-making process. Fig. 1 illustrates the overall architecture of the proposed system.

1) Task submission and request manager: The "Task Submission and Request Manager" acts as the entry point of the system. It receives task requests submitted by cloud users along with their associated parameters such as arrival time, computational size, deadline, and priority. Upon submission, tasks are validated and forwarded to the scheduling and decision-making components for further processing. This module ensures that user requirements are correctly captured and passed to the system without modification.

2) Resource monitoring module: This module continuously observes the operational status of physical hosts and VMs within the cloud data center. It collects real-time and

historical information related to CPU utilization, memory usage, bandwidth availability, energy consumption, and thermal conditions. In addition, this module tracks VM reliability indicators such as failure history, utilization trends, and anomaly signals. The collected data is periodically updated and made available to the decision-making engine to support accurate and timely fault tolerance decisions.

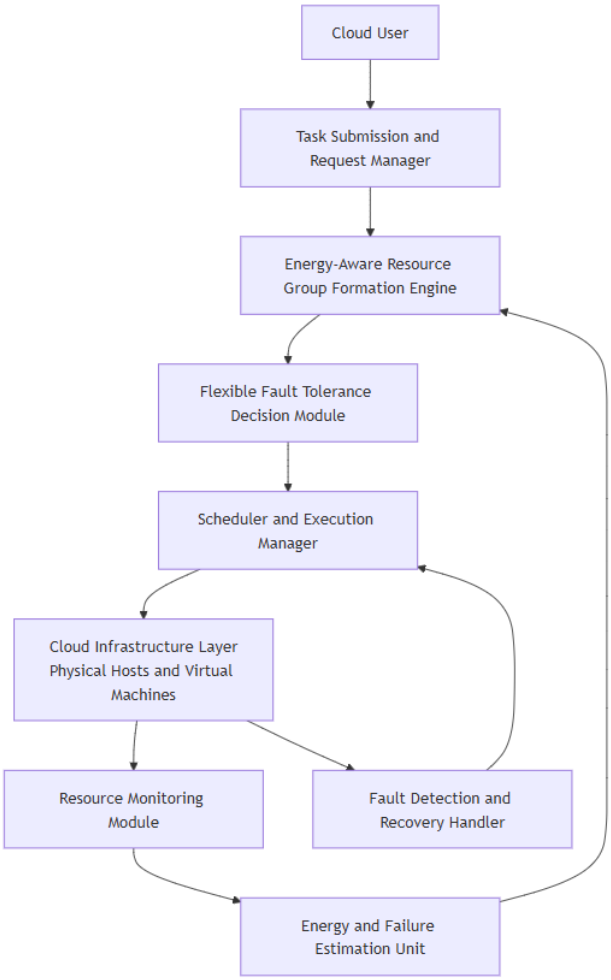

Fig. 1. Proposed system architecture.

3) Energy and failure estimation unit: This unit processes raw monitoring data to estimate energy consumption and failure probabilities. Energy estimation is performed using utilization-based power models at the host level, enabling the system to quantify the energy cost of task execution, VM migration, and fault tolerance operations. Failure estimation leverages reliability indicators to compute the probability of VM failures. These estimations form the basis for evaluating the trade-offs between reliability improvement and energy overhead.

4) Energy-aware resource group formation engine: The "Energy-Aware Resource Group Formation Engine" is the core intelligence of the proposed system. Using the multi-objective optimization approach, this engine evaluates available VMs based on energy efficiency, reliability, and resource suitability. It dynamically forms an optimal resource group for each task or task set, ensuring that selected resources satisfy deadline and reliability constraints while minimizing energy consumption. This component enables adaptive and context-aware fault tolerance decisions.

5) Flexible fault tolerance decision module: This module determines the most appropriate fault tolerance strategy for each task. Based on inputs from the resource group formation engine, this module selects between replication, checkpointing, VM migration, or hybrid fault tolerance techniques. The selection is influenced by task criticality, failure risk, expected energy cost, and execution deadlines. By avoiding static fault tolerance policies, this module ensures efficient and energy-conscious fault handling. The decision module follows a fixed priority order. If multiple VMs are assigned to the task, replication is selected. Otherwise, checkpointing is preferred for computationally intensive tasks (e.g., tasks exceeding the predefined workload threshold). VM migration is triggered when the hosting VM exceeds the utilization threshold. In all other cases, the framework employs the hybrid strategy.

6) Scheduler and execution manager: The "Scheduler and Execution Manager" is responsible for mapping tasks to the selected VMs and managing their execution lifecycle. It initiates task execution, coordinates checkpointing operations, and triggers VM migration or task replication when faults are predicted or detected. This module ensures that scheduling decisions comply with system constraints and that task execution progresses smoothly even under failure conditions.

7) Fault detection and recovery handler: This component continuously monitors task execution and VM health to identify failures or abnormal behavior. Upon detecting a fault or receiving a failure prediction signal, this component activates the recovery process defined by the selected fault tolerance strategy. Recovery actions may include switching to replica tasks, restoring from checkpoints, or migrating tasks to alternative VMs within the resource group. This module plays a critical role in maintaining service continuity.

8) Cloud infrastructure layer: The "Cloud Infrastructure Layer" consists of the physical hosts and VMs that provide computational, storage, and networking resources. This layer executes user tasks and supports dynamic VM provisioning and migration. It also serves as the source of monitoring data required for energy estimation and fault prediction. The infrastructure layer enables the practical deployment of the flexible fault tolerance system in real or simulated cloud environments.

VI. Results and Discussion

This section presents the experimental results obtained from simulation-based evaluation of the proposed framework. The experiments are designed to assess the system's ability to balance energy efficiency with fault tolerance effectiveness under realistic cloud workload conditions. To place the results in context, the proposed approach is compared against two baseline scheduling strategies: Best Fit, which selects VMs

greedily by reliability without regard for energy consumption, and Min Energy, which prioritizes the lowest-energy VMs without enforcing any reliability constraints. Together, these baselines represent opposite ends of the energy–reliability spectrum, making them appropriate reference points for evaluating the trade-off achieved by the proposed system.

A. Experimental Setup

The simulation environment consists of 20 VMs distributed across five physical hosts. Each VM is randomly configured with a CPU capacity ranging from 1,000 to 5,000 MIPS, memory between 4 GB and 16 GB, bandwidth between 100 and 1,000 Mbps, and an idle power draw between 50 W and 100 W, with a peak consumption of up to 300 W. VM failure probabilities are sampled uniformly in the range [0.01, 0.15], and reliability values are derived accordingly. Physical hosts operate with idle power between 200 W and 400 W and peak power up to 1,500 W.

A total of 500 tasks are generated for each simulation run. Task arrival times are distributed uniformly over a 100-time-unit window, with instruction counts ranging from 1,000 to 50,000 MI and relative deadlines between 10 and 100 time units. Each task carries a minimum reliability requirement drawn uniformly from [0.90, 0.999], reflecting the heterogeneous QoS demands typical of real cloud workloads. All three algorithms are evaluated on the identical task and VM set, generated using a fixed random seed to ensure a fair and reproducible comparison. In all experiments, the weight parameters were empirically set to $\alpha = 0.6$ and $\beta = 0.4$. A slightly higher weight was assigned to the energy component because the primary objective of the proposed framework is to reduce energy overhead while maintaining acceptable reliability. For the flexible fault tolerance decision, workload threshold and utilization threshold were set to 10,000 MI and 0.8, respectively. The simulation runs for 200 time units with a discrete step of 1 time unit.

Four performance metrics are used for evaluation: task completion rate, total energy consumption (Wh), and average achieved reliability. The fault tolerance overhead is modelled by applying mechanism-specific energy multipliers of 1.1× for checkpointing, 1.2× for replication, 1.25× for the hybrid strategy, and 1.3× for VM migration.

B. Overall Performance Comparison

Table I summarizes the aggregate performance of the three algorithms across all evaluation metrics. Fig. 2 to Fig. 4 trace each metric over the course of the simulation.

TABLE I. PERFORMANCE COMPARISON OF THE PROPOSED SYSTEM AND BASELINE ALGORITHMS

Metric	Proposed System	Best Fit	Min Energy
Task Completion Rate (%)	97.60	100.00	79.20
Total Energy Consumed (Wh)	201.70	206.66	163.68
Avg. Energy per Task (Wh)	0.4034	0.4133	0.3274
Avg. Achieved Reliability	0.9841	0.9825	0.9941
Reliability Threshold Met (%)	98.57	99.40	94.95

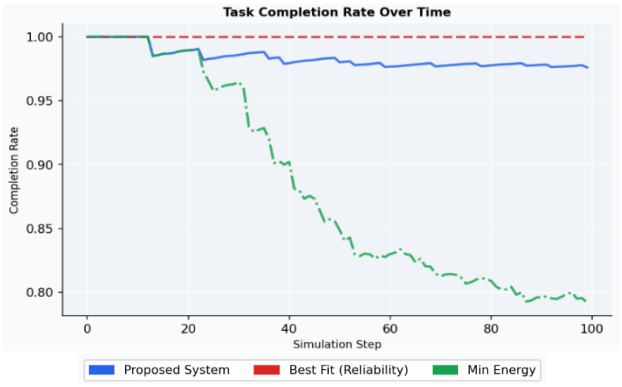

Fig. 2. Task completion rate over time.

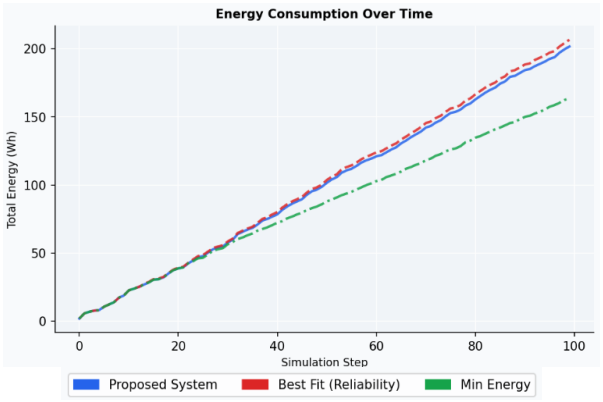

Fig. 3. Energy consumption over time.

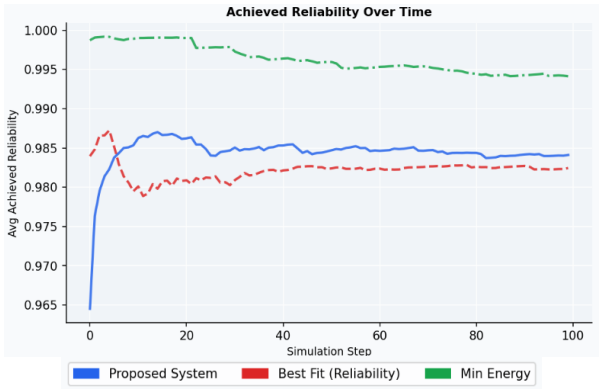

Fig. 4. Reliability over time.

1) Task completion analysis: Considering task completion rate (Fig. 2), the Best Fit baseline achieves a perfect completion rate of 100%, scheduling all 500 tasks within their respective deadlines. The proposed framework completes 488 tasks, corresponding to a 97.60% completion rate, while Min Energy completes only 396 tasks (79.20%). The notably low completion rate of Min Energy is a direct consequence of its energy-first selection policy, which occasionally assigns tasks to VMs with low processing capacity or high failure probability, resulting in missed deadlines and execution failures. The Best Fit baseline avoids this pitfall by unconditionally prioritising high-reliability VMs; however, as discussed below, this comes at a measurable energy cost.

2) Energy consumption analysis: Fig. 3 shows that Min Energy records the lowest total energy at 163.68 Wh, which is expected given that the algorithm explicitly optimizes for this metric. However, this apparent efficiency is somewhat misleading because Min Energy fails to complete about 20% of the tasks; a large fraction of potential execution energy is simply never expended. When energy is evaluated on a per-task basis, Min Energy consumes 0.3274 Wh per task, compared with 0.4034 Wh for the proposed system and 0.4133 Wh for Best Fit. The difference between the proposed system and Best Fit on this metric (approximately 2.4%) reflects the energy savings achieved by incorporating energy awareness into the VM selection process without sacrificing task throughput.

The Best Fit baseline incurs the highest total energy (206.66 Wh) precisely because it routes all tasks to the most reliable VMs irrespective of their power characteristics. High-reliability VMs tend to be more heavily provisioned and therefore draw more power. The proposed system avoids this by balancing energy cost and reliability through the composite scoring function, which avoids the selection of power-consuming VMs when more energy-efficient alternatives offer comparable reliability.

3) Reliability analysis: All three algorithms achieve high average reliability for the tasks they complete (Fig. 4). Min Energy records the highest average achieved reliability at 0.9941, which appears to be confusing at first observation. However, this result reflects a survivorship effect. Min Energy preferably selects low-power VMs; many of those may have low failure probabilities, and the tasks it completes tend to be those that benefit most from this selection. The proposed framework achieves an average reliability of 0.9841, and Best Fit achieves 0.9825. The small difference is largely attributed to the replication fallback mechanism in the proposed framework, which adds VMs to the resource group when the reliability threshold is not immediately satisfied by the top-ranked candidate.

A more practical reliability indicator is the proportion of completed tasks for which the achieved reliability met or exceeded the task-specific minimum threshold (Table I). Our proposed system satisfies this constraint for 98.57% of its completed tasks, compared with 99.40% for Best Fit and only 94.95% for Min Energy. This ordering is consistent with the design intent of each algorithm: Best Fit is most conservative about reliability, Min Energy is least so, and the proposed system occupies an intermediate position that trades a marginal reduction in reliability constraint satisfaction for a meaningful reduction in energy overhead.

C. Discussion

The obtained simulation results support the primary claim of this study that simultaneous optimization of energy efficiency and fault tolerance reliability is both feasible and beneficial. The proposed energy-aware flexible fault tolerance system demonstrates that the two objectives are not inherently opposed. A carefully designed composite scoring function, combined with an adaptive fault tolerance selection policy, can achieve near-complete task execution (97.60%). Also, the energy consumption is only marginally above the unreliable Min Energy baseline and meaningfully below the Best Fit baseline.

It should be noted that the Best Fit baseline achieves a 100% completion rate in these experiments. This is because the simulated workload, while diverse, does not saturate the VM pool to the point where reliability-first selection creates resource contention. In more congested scenarios, where the highest-reliability VMs are already heavily loaded, the Best Fit strategy would be expected to incur greater deadline misses and higher energy costs simultaneously, which would further strengthen the case for the balanced approach adopted by the proposed framework. Similarly, Min Energy's poor completion rate would be expected to worsen under stricter deadline distributions. It reinforces that energy minimization without reliability awareness is unsuitable as a standalone policy in fault-tolerant cloud environments.

The results also highlight an important practical consideration. The 2.40% deadline miss rate of the proposed framework consists primarily of tasks whose reliability requirements were sufficiently stringent with respect to the replication limit while satisfying the deadline constraint. Relaxing the replication limit or introducing additional VMs into the pool would be expected to reduce this residual miss rate further, at a modest increase in energy consumption. This configurability is an inherent strength of the proposed framework.

VII. Conclusion and Future Research Scope

Cloud service providers increasingly face the dual challenge of maintaining high reliability while controlling the growing energy demands of large-scale data centers. Although fault tolerance remains essential for ensuring service continuity, many existing approaches achieve reliability at the cost of significant energy overhead. This study addresses this challenge by proposing an intelligent energy-aware flexible fault tolerance system that integrates reliability and energy efficiency into a unified decision-making framework.

The proposed approach combines an energy-aware resource group formation algorithm with adaptive fault tolerance selection, enabling the system to dynamically choose suitable resources and fault tolerance mechanisms according to workload characteristics, reliability requirements, and system conditions. By incorporating both energy consumption and reliability metrics into the resource selection process, the framework effectively balances two traditionally conflicting objectives. The mathematical model, optimization strategy, and modular system architecture collectively provide a practical foundation for implementing sustainable fault-tolerant cloud environments.

Simulation-based evaluation demonstrated the effectiveness of the proposed system. The framework completed 97.60% of submitted tasks while satisfying reliability requirements for more than 98% of completed executions. At the same time, it reduced overall energy consumption compared with a reliability-focused baseline, illustrating that energy-aware fault

tolerance can be achieved without substantially compromising service quality. These findings confirm that intelligent resource selection and adaptive fault tolerance policies can significantly improve the operational efficiency of cloud infrastructures.

For future research, firstly, the proposed approach will be validated using large-scale cloud simulation platforms and real-world cloud workloads to further assess its scalability, adaptability, and effectiveness under highly dynamic operating conditions. Secondly, the principles of quantum computing will be incorporated into the scheduling decisions to study their effectiveness in a cloud computing environment. Performance comparison with the state-of-the-art frameworks will also be considered.

DECLARATION ON GENERATIVE AI

Several GenAI tools were used to complete this research work. ChatGPT, Claude, and DeepSeek were used for language refinement and grammar correction. Mermaid Live Editor was used to draw Fig. 1 (Proposed system architecture).

REFERENCES

[1] M. Hogan, F. Liu, A. Sokol, and J. Tong, "NIST Cloud Computing Standards Roadmap," 2013. doi: 10.6028/NIST.SP.500-291r2.

[2] G. Bosilca, R. Delmas, J. Dongarra, and J. Langou, "Algorithm-based fault tolerance applied to high performance computing," J. Parallel Distrib. Comput., vol. 69, no. 4, pp. 410–416, 2009, doi: 10.1016/j.jpdc.2008.12.002.

[3] M. Hasan and M. S. Goraya, "Fault tolerance in cloud computing environment: a systematic survey," Comput. Ind., vol. 99, pp. 156–172, 2018, doi: 10.1016/j.compind.2018.03.027.

[4] A. Safari, H. Sorouri, A. Rahimi, and A. Oshnoei, "A systematic review of energy efficiency metrics for optimizing cloud data center operations and management," Electronics (Switzerland), vol. 14, no. 11. 2025. doi: 10.3390/electronics14112214.

[5] T. Nazare, J. Gadelha, E. Nepomuceno, and R. Lozi, "Green computing for energy transition: a survey," IEEE Lat. Am. Trans., vol. 21, no. 9, pp. 937–948, 2023, doi: 10.1109/TLA.2023.10251799.

[6] S. Singh, R. Kumar, and D. Singh, "An empirical investigation of task scheduling and VM consolidation schemes in cloud environment," Computer Science Review, vol. 50. 2023. doi: 10.1016/j.cosrev.2023.100583.

[7] J. Samual, M. Hussin, N. A. W. A. Hamid, and A. Abdullah, "Frequency aware task scheduling using DVFS for energy efficiency in Cloud data centre," Expert Systems, vol. 42, no. 1. 2025. doi: 10.1111/exsy.13276.

[8] M. Hasan, M. S. Goraya, and T. Garg, "E-FFTF : An extended framework for flexible fault tolerance in cloud," in IoT and Analytics for Sensor Networks. Lecture Notes in Networks and Systems, 2021, pp. 35–45. doi: 10.1007/978-981-16-2919-8_4.

[9] A. Patil and R. Patil, "Proactive and dynamic load balancing model for workload spike detection in cloud," Meas. Sensors, vol. 27, pp. 1–8, 2023, doi: 10.1016/j.measen.2023.100799.

[10] B. K. Ray, A. Saha, S. Khatua, and S. Roy, "Proactive fault-tolerance technique to enhance reliability of cloud service in cloud federation environment," IEEE Trans. Cloud Comput., vol. 10, no. 2, pp. 957–971, 2022, doi: 10.1109/TCC.2020.2968522.

[11] A. Ragmani, A. Elomri, N. Abghour, K. Moussaid, M. Rida, and E. Badidi, "Adaptive fault-tolerant model for improving cloud computing performance using artificial neural network," Procedia Comput. Sci., vol. 170, pp. 929–934, 2020, doi: 10.1016/j.procs.2020.03.106.

[12] P. Senevirathne, S. Cooray, J. D. Herath, and D. Fernando, "Virtual machine proactive fault tolerance using log-based anomaly detection," IEEE Access, vol. 12, pp. 178951–178970, 2024, doi: 10.1109/ACCESS.2024.3506833.

[13] P. Kumari and P. Kaur, "An adaptable approach to fault tolerance in cloud computing," Int. J. Cloud Appl. Comput., vol. 13, no. 1, pp. 1–24, 2023, doi: 10.4018/IJCAC.319032.

[14] S. M. Umar, S. Sheikh, and S. M. Idrees, "Enhanced priority based task scheduling with integrated fault tolerance in distributed systems," Int. J. Cogn. Comput. Eng., vol. 6, pp. 152–169, 2025, doi: 10.1016/j.ijcce.2024.12.006.

[15] C. K. Dehury, P. K. Sahoo, and B. Veeravalli, "RRFT : A rank-based resource aware fault tolerant strategy for cloud platforms," IEEE Trans. Cloud Comput., vol. 11, no. 2, pp. 1257–1272, 2023, doi: 10.1109/TCC.2021.3126677.

[16] S. U. Mushtaq, S. Sheikh, A. Nain, S. Bharany, R. M. Ghoniem, and B. M. Taye, "CRFTS : a cluster-centric and scheduling strategy to enhance QoS in cloud computing," Sci. Rep., vol. 15, pp. 1–21, 2025.

[17] S. U. Mushtaq, S. Sheikh, and S. M. Idrees, "Next-gen cloud efficiency: Fault-tolerant task scheduling with neighboring reservations for improved resource utilization," vol. 12, no. May, 2024.

[18] M. K. Roberts, S. K. Shanmugam, M. Al-Razgan, and T. Alfakih, "A nature-inspired dual phased fuzzy hybrid algorithm for adaptive self-healing and dynamic energy-aware resource management in cloud computing," J. Cloud Comput., vol. 14, no. 62, pp. 1–23, 2025.

[19] S. Gore, Y. Bhapkar, J. Ghadge, S. Gore, and S. K. Singha, "Evolutionary programming for dynamic resource management and energy optimization in cloud computing," in International Conference on Advanced Computing Technologies and Applications (ICACTA), IEEE, 2023. doi: 10.1109/ICACTA58201.2023.10393769.

[20] S. Selvaraju, S. Maurya, Z. Adilov, S. Akhmedov, S. P. K. Gudla, and G. M, "Developing a carbon neutral cloud infrastructure with intelligent power management in green cloud AI using reinforcement learning," in International Conference on Emerging Technologies in Engineering Applications (ICETEA), IEEE, 2025. doi: 10.1109/ICETEA64585.2025.11099898.

[21] S. R. Jena, U. Srilakshmi, and S. M. Rafi, "Intelligent green energy solutions : Optimizing renewable energy for edge cloud computing," in 2025 5th International Conference on Pervasive Computing and Social Networking (ICPCSN), IEEE, 2025, pp. 765–769. doi: 10.1109/ICPCSN65854.2025.11035813.

[22] A. Khan, F. Ullah, D. Shah, M. H. Khan, S. Ali, and M. Tahir, "EcoTaskSched : a hybrid machine learning approach for energy-efficient task scheduling in IoT- based fog-cloud environments," Sci. Rep., vol. 15, pp. 1–27, 2025.

[23] J. Ramesh, Z. Solatidehkordi, K. El-fakih, and R. Aburukba, "Minimizing virtual machine live migration latency for proactive fault tolerance using an ILP model with hybrid genetic and simulated annealing algorithms," IEEE Access, vol. 12, pp. 107232–107246, 2024, doi: 10.1109/ACCESS.2024.3438358.

[24] Y. I. Lysevych and V. E. Volkov, "Mathematical model for optimizing cloud IT infrastructure," Informatics. Cult. Technol., vol. 2, pp. 237–241, 2025, doi: 10.15276/ict.02.2025.36.

[25] B. Fei, X. Zhu, D. Liu, J. Chen, W. Bao, and L. Liu, "Elastic resource provisioning using data clustering in cloud service platform," IEEE Trans. Serv. Comput., vol. 15, no. 3, pp. 1578–1591, 2022, doi: 10.1109/TSC.2020.3002755.

[26] "What is a Virtual Machine? | Microsoft Azure." Accessed: Jul. 18, 2026. [Online]. Available: https://azure.microsoft.com/en-us/resources/cloud-computing-dictionary/what-is-a-virtual-machine

[27] C. Liu and S. Baskiyar, "A general distributed scalable grid scheduler for independent tasks," J. Parallel Distrib. Comput., vol. 69, no. 3, pp. 307–314, 2009, doi: 10.1016/j.jpdc.2008.11.003.

[28] J. Wang, W. Bao, X. Zhu, T. Yang, and Y. Xiang, "FESTAL: fault-tolerant elastic scheduling algorithm for real-time tasks in virtualized cloud," IEEE Trans. Comput., vol. 64, no. 9, pp. 2545–2558, 2015, doi: 10.3969/j.issn.1000-436x.2014.10.020.

[29] L. Ismail and E. H. Abed, "Linear power modeling for cloud data centers: taxonomy, locally corrected linear regression, simulation framework and evaluation," IEEE Access, vol. 7, pp. 175003–175019, 2019, doi: 10.1109/ACCESS.2019.2956881.