WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

A Mechanism for Cooperative Demand-Side Management

Miss. Ashiyana Pathan[1], Dr. Hirendra R. Hajare[2]

M-Tech Scholar, Computer Science and Engineering, Ballarpur Institute of Technology, Ballarpur, Maharashtra, India[1]
Assistant professor and HOD, Computer Science and Engineering, Ballarpur Institute of Technology, Ballarpur, Maharashtra, India[2]

ABSTRACT :

With the proliferation of cloud computing and the growing demand for real-time, energy-efficient digital infrastructure, cooperative Demand-Side Management (DSM) has emerged as a vital strategy. This paper proposes a fault-aware and trust-based load balancing mechanism to optimize DSM within distributed cloud systems and smart grids. The model combines blockchain technology and machine-to-machine (M2M) communication for secure, autonomous coordination of computing and storage resources. Simulation experiments show considerable gains in fault tolerance, response time, system utilization, and request success rates. Thse DSM mechanism leverages the use of smart contracts to facilitate automatic task negotiation, trust models for scheduling reliability, and load balancing algorithms to distribute workloads across cloud infrastructure dynamically. Through tackling issues like non-uniform load distribution, fault rates, and transparency in the conventional DSM systems, the introduced mechanism makes a distributed computing system more efficient and reliable. CloudSim and MATLAB were used for simulation, where the focus was on parameters like resource utilization, fault rates, and response time.

Keywords: : Demand-Side Management, Fault Tolerance, Load Balancing, Blockchain, Smart Contracts, Trust Management, Cloud Computing, M2M Communication.

INTRODUCTION

The development of intelligent ecosystems and intelligent grid infrastructure demands a revolution in the way computer and power resources are controlled. Cloud computing DSM systems and power grids are prone to numerous inefficiencies, including centralized control, no fault tolerance, and lack of scalability. These inefficiencies result in bottlenecks during performance, particularly at high load levels, degrading system dependability and rising task and service failure probability.

Contemporary cloud environments suffer from uncertain workloads and random component failures, which require mechanisms to adapt in real time. There is also increasing need for decentralized systems that can work with minimal human oversight while maintaining security, transparency, and performance. This has prompted the development of blockchain and M2M communication as anchor technologies for filling the gaps.

To address these issues, this work proposes a new DSM approach that combines blockchain-based smart contracts with a fault-tolerant and trust-aware scheduling framework. The approach enables distributed cloud servers and microgrid nodes to manage and balance their own workloads independently while ensuring transparency and low overhead. With the use of both trust measures and resource factors, the system guarantees secure task allocation, resulting in enhanced quality of service (QoS) and operational robustness.

LITERATURE SURVEY

There has been a vast amount of research on load balancing, resource allocation, and DSM in cloud and distributed systems. Round-Robin and Min-Min algorithms have been popular task allocation algorithms because they are simple but tend to ignore system heterogeneity and dynamic characteristics, thus giving suboptimal outcomes [1][2]. Max-Min and Honeybee algorithms enhance task allocation using heuristic techniques but are not very reliable when nodes fail [3][4].

Metaheuristic methods such as Particle Swarm Optimization and Genetic Algorithms have better adaptability with increased computational complexity and convergence time [5][6]. ACO-based scheduling models have better resource mapping due to imitating biological behavior, but their performance becomes poor in large networks [7]. Predictive load balancing frameworks using history and real-time information have proven effective in lowering request failure rates [8].

Power efficiency and cost savings in resource scheduling have been highlighted by research such as TESA and GreenMonster, which minimize server loads for energy conservation [9][10]. MaxUtil and power-conscious scheduling methods also seek to minimize operational costs, but at the expense of fault resilience as an important metric [11].

Trust management systems have been proposed to enhance reliability in multi-agent systems. Models introduced in [12][13] analyze past behavior and trust to prevent malicious or underperforming nodes. But most are constrained to SaaS layers and don't provide a solution for infrastructure-level trust, which is imperative in heterogeneous settings.

Studies have recently investigated the combination of blockchain and DSM. Wu et al. [14] employed a DSM framework with MultiChain, which allowed contract execution in a secure manner. Yuan et al. [15] proposed a Bayesian incentive mechanism for DSM at the neighborhood level to promote collaborative energy consumption. Kouhestani et al. [16] designed a blockchain-enabled peer-to-peer energy trading platform, improving energy trade transparency and fairness.

Pop et al. [17] discussed the role of smart contracts in decentralized energy markets, where they have the capability to lower transaction costs. Li et al. [18] concentrated on the integration of blockchain and IoT with smart grids, where they faced scalability challenges. Goranovic et al. [19] proposed light-weight consensus protocols appropriate for DSM in energy-limited networks.

More studies highlight the necessity of edge computing and real-time interoperability. Research in [20][21] puts forth hybrid approaches merging fog computing and blockchain to minimize latency. Models in [22][23] recommend hierarchical control systems for DSM systems on a large scale. These works provide the basis upon which this research constructs its cooperative, blockchain-based DSM framework.

METHODOLOGY

The designed DSM mechanism consists of three embedded layers: (1) fault detection and load balancing, (2) trust management, and (3) blockchain-based execution of smart contracts. Each layer plays a unique role in facilitating cooperative use of distributed cloud resources.

3.1 Load Balancing and Fault Detection

The system utilizes two main algorithms: Fault and Load Based Load Balancing Algorithm (FLBLBA) and Deadline-Aware Load Balancing Algorithm (DDLBA). The two algorithms compare parameters like request queue length, CPU availability, and processing time. FLBLBA aims to reduce request failure by dynamically relocating loads to the least loaded and least faulty server. DDLBA takes into account task deadlines to allow real-time execution. Figures 3.3 and 3.10 of the thesis demonstrate the working steps of these algorithms.

3.2 Trust Management

Trust is computed using both direct (MIPS, RAM) and indirect (response time, fault rate) parameters. The trust is initially calculated by static hardware features, and dynamic trust is updated based on successful task execution. The system can thereby choose the most trusted servers, especially for private or sensitive tasks. Equations 3.19 and 3.20 demonstrate how the trust values are calculated.

3.3 Blockchain Integration

With MultiChain, a permissioned blockchain framework, the system uses smart contracts for task bidding, negotiation, and settlement. When a server is overloaded, smart contracts take charge to negotiate with peer nodes to transfer tasks autonomously. Atomic transactions see to it that task transfers along with compensations are done simultaneously. The blockchain is also an immutable ledger of all transactions that enables full traceability and auditability.

RESULTS AND DISCUSSION

Table 1: FLBLBA – Simulation Environment Parameters

No. of Client Requests	No. of Servers	CPU Cores per Server	Storage (GB)	Server Queue Length
800	12	7	500	20 1000
1200	12	7	500	25
1800	12	7	500	25
2400	12	7	500	25

The Table 1 outlines the test scenarios used for the **Fault and Load Based Load Balancing Algorithm (FLBLBA)**. Each simulation run varied the number of client requests from 800 to 2400 to assess how the algorithm performs under different workload intensities. Across all experiments:

- **12 servers** were consistently used.
- Each server was equipped with **7 CPU cores**, **500 GB of storage**, and queue lengths were adjusted (from 20 to 25) to match increasing loads.

The purpose of this configuration was to evaluate how well FLBLBA handles fault-aware task distribution under growing pressure while avoiding server

overloads.

Table 2: DDLBA – Simulation Environment Parameters

No. of Client Requests	No. of Servers	CPU Cores per Server	Storage (GB)	Server Queue Length
800	12	7	500	15 1000
1200	12	9	500	20
1800	12	10	500	20
2400	12	11	500	20

In contrast, the **Deadline-Aware Load Balancing Algorithm (DDLBA)** simulation emphasized time-sensitive task execution. As the number of client requests increased, the system scaled CPU cores per server from 7 to 11 to simulate an elastic cloud environment. Queue lengths were also varied slightly to maintain fairness and prevent excessive delay.

DDLBA was tested under the assumption that certain client requests have critical deadlines. Hence, the algorithm had to prioritize these tasks based on their urgency while minimizing missed deadlines or task failures.

Table 3: Server Parameters for Trust-Based Scheduling

Server	Fault Rate	Hard Disk (MB)	RAM (MB)	MIPS	Cores
Server1	**0.143**	**1,000,000**	**20048**	**10000**	**4**
Server2	**0.125**	**1,000,000**	**20048**	**10000**	**6**
Server3	**0.500**	**1,000,000**	**20048**	**10000**	**4**

This table shows the heterogeneous nature of the server environment, emphasizing the trust computation aspect of the proposed DSM model. Key parameters used in the trust formula include:

- Fault Rate: Indicates the likelihood of server failure. For example, Server3 has a high fault rate of 0.5, making it less preferable.
- RAM and MIPS: These indicate the processing capability of servers.
- Cores: More cores mean better multitasking capability.

Servers with lower fault rates and higher processing capabilities were assigned higher trust scores and prioritized for task scheduling, ensuring reliability and performance.

Table 4: Performance Summary (DDLBA vs Least Loaded)

Metric	DDLBA	Least Loaded
Completed Requests	**Higher**	**Lower**
Failed Requests	**Lower**	**Higher**
Average Response Time	**Lower (IMPROVED)**	**Higher**
System Utilization	**Higher**	**Lower**
Request Postponement Count	**Lower**	**Higher**

This comparison highlights the performance difference between DDLBA and a conventional Least-Loaded scheduling algorithm. Key observations include:

- DDLBA completed more requests than the Least-Loaded method, especially under heavy loads.
- Failed requests were fewer due to deadline-based prioritization and dynamic task redistribution.
- Average response times were lower, proving the system's ability to respond swiftly to high-priority requests.
- System utilization improved, reflecting efficient use of server resources.
- Fewer requests were postponed, indicating better scheduling and load balancing strategies.

CONCLUSION

This work proposes a secure and decentralized framework for collaborative Demand-Side Management (DSM) in distributed cloud and smart grid systems. With the combination of blockchain-based smart contracts, trust-aware scheduling, and adaptive load balancing algorithms, the system mitigates major limitations of traditional DSM models, such as centralization, absence of fault tolerance, and inadequate scalability. Through CloudSim and MATLAB-based simulations, the system proved substantial enhancement in task rates of completion, utilization, and response time while reducing failure and bottleneck events under changing workloads.

Usage of Fault and Load Based Load Balancing Algorithm (FLBLBA) and Deadline-Driven Load Balancing Algorithm (DDLBA) and real-time trust analysis enables distributed agents to make smart and autonomous choices. Utilizing a permissioned blockchain guarantees transparency and safe coordination without centralized control. Although the outcomes are encouraging, more work is needed to solve blockchain latency, consensus overhead, and deployment at scale. Future research will involve incorporating real-time IoT data, investigating hybrid consensus protocols, and compatibility with current legacy infrastructure to enable wider adoption in energy and cloud computing spaces.

REFERENCES

[1] Ullah, F. et al. (2018). A comparison of performance in scheduling algorithms within cloud computing.
[2] Wang, L. et al. (2017). Optimal cloud resource scheduling using min-min algorithm optimized.
[3] Xu, C. et al. (2016). Max-min optimized by trust factors.
[4] Tang, M. et al. (2015). Fault-aware scheduling inspired by honeybees.
[5] Lee, Y.C. et al. (2012). Genetic algorithms applied to multi-objective resource allocation.
[6] Kennedy, J. et al. (1995). Particle swarm optimization.
[7] Dorigo, M. et al. (2006). Ant Colony Optimization for task scheduling.
[8] Zhang, Q. et al. (2016). Predictive scheduling with historical data.
[9] Wang, J. et al. (2017). TESA: Three-threshold energy saving algorithm.
[10] Li, H. et al. (2018). GreenMonster: A holistic power-aware scheduler.
[11] Kim, K. et al. (2020). MaxUtil for cloud efficiency.
[12] Josang, A. et al. (2007). A survey of trust models in distributed systems.
[13] Noor, T.H. et al. (2013). Trust-based service selection.
[14] Wu, J. et al. (2017). Blockchain for energy-efficient management in smart grids.
[15] Yuan, Y. et al. (2017). Enki: A cooperative Bayesian incentive model for DSM.
[16] Kouhestani, M. et al. (2020). P2P energy trading platform using smart contracts.
[17] Pop, C. et al. (2018). A review on smart contracts in energy markets.
[18] Li, X. et al. (2019). IoT and blockchain integration challenges in smart grids.
[19] Goranovic, A. et al. (2017). Lightweight consensus for energy-constrained DSM.
[20] Mahmud, R. et al. (2019). Edge computing and blockchain integration.
[21] Bera, B. et al. (2020). Interoperable blockchain architectures.
[22] Rahman, A. et al. (2018). Hybrid fog-cloud systems.
[23] Yang, C. et al. (2021). Hierarchical DSM frameworks for smart grids.