An Energy- and SLA-Aware Heuristic for Efficient Virtual Machine Migration and Resource Scheduling in Cloud Computing

C. S. Prasanna[1], Dr. Mayan Parihar Singh[2]

[1]Research Scholars, Department of IT, Dr. C.V. Raman University
[2]Assistant Professor, Department of IT, Dr. C.V. Raman University

Abstract
Cloud computing provides elastic access to virtualized computing resources, but efficient scheduling remains difficult because workloads, resource availability, and service requirements change continuously. Conventional scheduling methods can reduce execution time for selected workloads, yet they may not simultaneously address resource utilization, service-level agreement (SLA) compliance, virtual machine (VM) migration, and energy consumption. This study develops and evaluates a heuristic resource-management approach that integrates host overload/underload detection, VM selection based on migration time, controlled VM migration, and host shutdown. The proposed approach uses CPU utilization as a principal indicator of host state and applies a migration-oriented decision process to reduce unnecessary movement while maintaining service continuity. The research is grounded in a CloudSim-based simulation framework and compares the proposed policy with established threshold- and median-absolute-deviation-based VM selection and migration policies. Results reported in the thesis show that the proposed approach achieved an overall SLA violation of 27.5%, compared with 38.2% for the referenced existing approach, representing a relative reduction of approximately 28.0%. The proposed method also recorded 16 SLA-related host shutdown events in the detailed experiment, while the compared methods generally recorded 17 or more, and the thesis reports lower energy consumption than the evaluated heuristic policies. A separate baseline experiment in the thesis also confirms that Min-Min provides the shortest execution time among FCFS, SJF, Round Robin, Max-Min, and Min-Min under the stated CloudSim configuration. The findings indicate that scheduling decisions should be multi-objective rather than based on execution time alone. The proposed heuristic is therefore positioned as a practical, simulation-validated direction for energy-aware and SLA-conscious cloud resource management.

Keywords: cloud computing; task scheduling; virtual machine migration; SLA violation; energy efficiency; resource management; CloudSim; heuristic scheduling

1. Introduction

Cloud computing has transformed the provisioning of computing resources by enabling users to obtain processing, storage, networking, and software services on demand. Virtualization is central to this model because physical resources can be represented as virtual machines (VMs), allowing multiple workloads to share a physical host. Although elasticity improves flexibility, it also creates a difficult resource-management problem: tasks and VMs must be mapped to heterogeneous resources while workloads fluctuate over time. The thesis underlying this article identifies resource allocation and task scheduling as critical challenges because inefficient scheduling can increase execution time, cost, energy consumption, load imbalance, network congestion, and quality-of-service (QoS) degradation. filecite turn1file0 L27-L47

Task scheduling is particularly challenging because the assignment problem in distributed and cloud systems is computationally difficult and because practical decisions must often balance several objectives. The thesis review emphasizes makespan, fairness, load balancing, deadline satisfaction, cost, energy, SLA compliance, and resource utilization as important scheduling considerations. These objectives may conflict. For example, aggressive consolidation can reduce the number of active hosts and energy consumption, but excessive migration can increase overhead and contribute to SLA degradation. Consequently, a useful scheduling policy must respond to the current state of hosts and VMs rather than relying exclusively on a static task-ordering rule.

Initial experiments reported in the thesis compare five conventional heuristics—First-Come, First-Serve (FCFS), Shortest Job First (SJF), Round Robin (RR), Max-Min, and Min-Min—using CloudSim 3.0.3. For 50, 100, 200, 300, and 400 cloudlets, Min-Min produced the shortest completion time in the reported experiment. However, the thesis explicitly notes that Min-Min does not fully address optimal resource utilization, deadlines, and energy efficiency. This observation motivates the development of scheduling methods that go beyond execution time alone. filecite turn1file2 L138-L156

The present article focuses on the thesis's later energy- and SLA-aware contribution: a heuristic that detects overloaded and underloaded hosts, selects VMs according to migration time, and places or consolidates VMs while controlling host shutdown. The objective is not to claim that one metric is universally optimal, but to demonstrate how a relatively simple heuristic can integrate resource state, migration overhead, SLA behavior, and energy considerations within a common scheduling framework.

2. Background and Related Work

2.1 Cloud Resource Scheduling

In a cloud environment, a task submitted by a user is ultimately executed on a virtualized resource. The scheduler must therefore obtain information about available resources, determine an appropriate VM or host, and allocate the task while satisfying service requirements. The thesis describes task scheduling as a process in which the scheduler observes the current state and attributes of accessible resources and assigns jobs to VMs according to task requirements. Effective scheduling should improve CPU utilization, reduce turnaround time, and increase throughput. filecite turn1file5 L346-L375

Classical heuristics remain useful as baselines because they are easy to implement and have relatively low computational overhead. FCFS prioritizes tasks according to arrival order; SJF prioritizes smaller jobs; Min-Min selects the task-resource combination with the smallest completion time; and Max-Min gives greater attention to large tasks. These approaches are valuable for understanding the execution-time dimension, but their decision criteria are not sufficient for energy-aware cloud management when the system must also control VM migration, host utilization, and SLA violations.

2.2 Energy-Aware and SLA-Aware Management

Energy consumption in data centers is closely related to the number and utilization level of active hosts. One strategy is consolidation: VMs can be moved from underutilized hosts to other suitable hosts so that an idle host can be switched off. The thesis review discusses energy-aware approaches based on VM allocation, consolidation, migration, and host status switching. It also notes that energy-aware scheduling must avoid excessive SLA violations and resource wastage. For example, the reviewed literature includes approaches using live migration, task consolidation, and metaheuristic optimization. filecite turn1file9 L586-L617

An SLA-aware policy must therefore distinguish between beneficial and harmful migration. Moving a VM can relieve an overloaded host, but migration consumes network and processing resources and may temporarily degrade service. The thesis identifies migration control as a key factor because an unsuitable resource mapping can overload the destination and increase the risk of SLA violation. In the proposed model, CPU utilization is used as the principal indicator for determining VM or host condition. filecite turn2file8 L589-L601

2.3 Research Gap

The thesis identifies four closely related gaps. First, existing scheduling approaches may not manage CSP resources effectively, causing underutilization or overloading. Second, many methods do not sufficiently reduce execution and response delay in dynamic environments. Third, cost and energy are often treated separately rather than jointly. Fourth, network bandwidth and memory receive limited attention even though both influence data transfer and resource allocation. These gaps suggest the need for a scheduling framework that combines resource-state awareness with performance, SLA, and energy objectives. filecite turn1file 7L477-L492

3. Objectives and Contributions

The study is aligned with the thesis objectives of efficient CSP resource management, reduced time delay, improved cost and energy behavior, and better management of network and memory resources. filecite turn1file4 L272-L284

- □Develop a host-state-aware heuristic for VM consolidation and migration.
- □Detect overloaded hosts using a utilization threshold and trigger corrective migration.
- □Identify underloaded hosts and consolidate their VMs so that the host can be deactivated when appropriate.
- □Select VMs with attention to migration time to limit the overhead of corrective actions.
- □Evaluate the approach using SLA violation, VM migration, host shutdown, and energy-related indicators.
- □Position execution-time scheduling and energy/SLA scheduling as complementary objectives rather than competing alternatives.

3.1 Main Contribution

The main contribution is a practical heuristic that links three decisions host overload/underload detection, VM selection and migration, and VM placement within one resource-management loop. Rather than attempting to solve the full scheduling problem through a computationally expensive global optimizer, the method makes local decisions based on current host utilization and migration cost. This design is intended to be suitable for simulation and potentially adaptable to real cloud orchestration systems.

4. Proposed Methodology

The proposed methodology is organized around the observation that host condition directly affects task performance, energy consumption, and SLA behavior. The algorithm continuously evaluates host utilization. When a host exceeds an upper utilization threshold, a VM is selected for migration using a minimum-migration-time policy. When a host is underloaded, its VMs are consolidated and the host is deactivated if the destination mapping remains feasible. The thesis defines four operational phases: host utilization assessment, overloaded-host migration, underloaded-host consolidation, and task execution. filecite turn2file8 L602-L610

4.1 Host Overload Detection

The overload module compares measured host utilization with an upper threshold of 0.90. If utilization exceeds this threshold, the host is classified as overloaded and VM migration is initiated. The thesis presents the overload procedure as a Boolean decision: measure host utilization, compare it with the upper threshold, and return True when the threshold is exceeded. filecite turn2file8 L613-L624

A normalized utilization representation can be expressed as:

$U_h = (\Sigma \text{ CPU demand of VMs on host } h) / (\text{CPU capacity of host } h)$

4.2 VM Selection and Migration

Once an overloaded host is identified, the scheduler chooses a VM that can be migrated with comparatively low migration time. This is important because migration itself introduces communication and processing overhead. The selected VM is transferred to an appropriate destination host, after which the source host is reevaluated. The decision is repeated only when necessary, which reduces unnecessary migration activity.

4.3 Underloaded Host Consolidation

For an underloaded host, the policy selects all VMs for consolidation and attempts to move them using the minimum-migration policy. Once the VMs have been relocated, the host can be deactivated. This step is intended to reduce the number of active hosts and consequently lower energy consumption. The method therefore combines performance protection for overloaded hosts with consolidation for underloaded hosts.

4.4 Overall Decision Process

Phase	Input	Decision	Expected effect
1. State assessment	Host/VM utilization	Classify host state	Identify overload or underload
2. Overload control	Overloaded host + candidate VMs	Select VM with low migration time	Reduce overload and SLA risk
3. Underload control	Underloaded host + resident VMs	Consolidate VMs	Enable host shutdown
4. Placement	Migrating VM + candidate hosts	Choose feasible destination	Maintain service continuity
5. Execution	Updated allocation	Run tasks	Measure SLA, energy and migration outcomes

4.5 Algorithmic Representation

Algorithm: Energy- and SLA-Aware VM Migration Heuristic
Input: Hosts H, VMs V, utilization threshold T=0.90
1. Measure utilization of each host $h \in H$.
2. If $U_h > T$, mark h as overloaded.
3. Select the VM requiring the smallest feasible migration time.
4. Select a destination host that can accept the VM without creating a new overload.
5. Migrate the selected VM and update host states.
6. If U_h is under the lower operating region, mark h as underloaded.
7. Move its VMs using the minimum-migration policy to feasible destinations.
8. If the host becomes empty, deactivate the host.
9. Execute pending tasks using the updated allocation.
10. Record SLA violations, VM migrations, host shutdowns and energy indicators.
Output: Updated VM-to-host mapping and performance metrics.

5. Experimental Design

The thesis uses CloudSim as the primary simulation framework. CloudSim was selected because it supports modelling of cloud infrastructure, virtual machines, tasks/cloudlets, data centers, scheduling policies, and resource allocation. The methodology chapter describes CloudSim as a generalized and extensible framework for modelling, simulation, and experimentation with cloud infrastructures and application services. filecite turn2file0 L47-L62

The earlier baseline experiment was conducted with CloudSim 3.0.3 at the VM level. The reported configuration used three VMs, 1024 MB RAM, 100,000 MB storage, and 10,000 Kbps bandwidth at the host level, while the VM configuration used 300 MIPS, 1,000 MB size, 512 MB RAM, and 1,000 Kbps bandwidth. The experiment compared FCFS, SJF, RR, Max-Min, and Min-Min across 50–400 cloudlets. Filecite turn1file2 L154-L188

Table 1. Baseline completion-time results reported in the thesis.

Cloudlet count	FCFS (s)	SJF (s)	RR (s)	Max-Min (s)	Min-Min (s)
50	80	71	66	60	56
100	103	90	82	80	77
200	166	155	151	136	134
300	210	202	193	188	187
400	238	230	218	215	211

For the proposed energy/SLA-aware evaluation, the thesis compares the proposed migration policy with multiple established host/VM selection combinations, including threshold-based and median-absolute-deviation-based policies. The reported indicators include SLA degradation due to migration, SLA time per active host, overall and average SLA violation, host shutdowns, VM migrations, and energy consumption. Because the source thesis does not provide every raw simulation parameter in the retrieved section, the present article reports only the configurations and outcomes explicitly documented in the source rather than inventing additional experimental settings.

6. Results and Analysis

6.1 Baseline Scheduling Result

The baseline results show a consistent ordering in which Min-Min records the lowest completion time among the five conventional heuristics. At 50 cloudlets, completion time decreases from 80 seconds under FCFS to 56 seconds under Min-Min. At 400 cloudlets, the corresponding values are 238 seconds and 211 seconds. Thus, the advantage of Min-Min is visible across all tested workload sizes. The thesis concludes that Min-Min outperformed the other basic methods in execution time under the simulated configuration. filecite turn1file1 L96-L108

Table 2. Derived relative reductions from the thesis-reported completion times.

Cloudlets	Min-Min vs FCFS reduction	Min-Min vs Max-Min reduction
50	30.0%	6.7%
100	25.2%	3.8%
200	19.3%	1.5%
300	11.0%	0.5%
400	11.3%	1.9%

6.2 Proposed Policy: VM Migration and SLA

The detailed experiment reported in the thesis shows that the proposed method produced an SLA degradation due to migration of 0.16%, compared with 0.17–0.18% for the listed comparison policies. It also recorded 6.32% SLA time per active host and 10.00% overall SLA violation in the detailed policy comparison. The proposed method recorded 16 host shutdowns, while the comparison methods generally recorded 17 or more. These results indicate that the proposed policy can maintain competitive SLA behavior while supporting consolidation and host deactivation. filecite turn2file3 L190-L234

Table 3. Policy-level results explicitly reported in the thesis.

Policy	SLA degradation due to migration	SLA time/active host	Overall SLA violation	Host shutdowns
Proposed	0.16%	6.32%	10.00%	16
thr_mmt	0.17%	6.24%	10.00%	17
thr_rs	0.17%	6.58%	10.00%	17

thr_mc	0.17%	6.24%	10.00%	17
thr_mu	0.17%	6.79%	10.00%	17
mad_mmt	0.17%	6.10%	10.00%	17
mad_rs	0.18%	6.72%	9.96%	19
mad_mc	0.17%	6.10%	10.00%	17
mad_mu	0.18%	6.51%	10.00%	18

6.3 Comparison with an Established Method

A second comparison reported in the thesis provides a direct existing-versus-proposed view. Overall SLA violation decreases from 38.20% in the established method to 27.5% under the proposed method. This corresponds to an approximate relative reduction of 28.0%. Average SLA violation remains 10.00% for both methods. The number of VM migrations changes from 832 to 840, an increase of approximately 0.96%. Therefore, the improvement in SLA behavior is achieved without a reduction in the number of migrations in this particular comparison. This trade-off is important and should be transparently reported rather than interpreted as universal superiority. filecite turn2file9 L647-L658

Table 4. Existing-versus-proposed comparison derived from thesis values.

Metric	Existing method	Proposed method	Relative change
Overall SLA violation	38.20%	27.50%	28.0% reduction
Average SLA violation	10.00%	10.00%	No change
VM migrations	832	840	0.96% increase

6.4 Energy Efficiency

The thesis reports that the proposed methodology consumes less energy than the evaluated heuristic methods and attributes this behavior to more effective management of VM migration, host shutdown, and resource consolidation. The reported graphs and statistics collectively indicate improved performance in VM migration, SLA violations, host shutdowns, and energy usage. filecite turn2file3 L228-L234

Because the retrieved thesis section does not expose the underlying numerical energy series, this article does not fabricate absolute energy values or percentage savings. For journal submission, the final manuscript should include the original energy table or the raw CloudSim output if the target journal requires reproducibility at the numerical level. This is particularly important for a Scopus-indexed venue, where transparent experimental reporting strengthens the credibility of the contribution.

7. Discussion

The experimental evidence supports two complementary conclusions. First, execution-time optimization can be effectively addressed by task-level heuristics such as Min-Min. Second, minimizing execution time alone is insufficient for a dynamic cloud infrastructure because a cloud provider must also control resource utilization, energy consumption, SLA violations, and migration overhead. The thesis itself identifies this limitation of Min-Min and motivates subsequent scheduling approaches that consider broader objectives. Filecite turn2file2 L147-L156

The proposed VM migration heuristic addresses this second problem at the infrastructure-management level. Its strength is conceptual simplicity: host state is measured, overload triggers corrective action, underload triggers consolidation, and migration is selected with attention to transfer time. Such a policy is easier to interpret than a black-box optimizer and may be attractive where predictable decision logic is required.

At the same time, the results reveal an important trade-off. In the direct existing-versus-proposed comparison, overall SLA violation falls substantially, but VM migrations increase slightly. This suggests that SLA improvement may require additional movement of VMs. The result does not invalidate the method; rather, it highlights the need for multi-objective evaluation. A future refinement could explicitly optimize a weighted objective containing SLA risk, migration cost, energy, and host utilization instead of treating migration time as the dominant local criterion.

The proposed 90% upper threshold is also a practical but relatively simple rule. Fixed thresholds are easy to implement, yet workload patterns differ across applications and time. Adaptive thresholds based on historical utilization, workload volatility, or predicted demand could improve robustness. Similarly, destination-host

selection could incorporate network bandwidth and memory availability more explicitly, addressing two dimensions identified in the thesis research gap.

8. Practical Implications

- Cloud service providers can use host-state monitoring to distinguish corrective migration from energy-saving consolidation.
- Data-center operators can evaluate scheduling policies using a combination of SLA, migration, host-shutdown, and energy indicators rather than completion time alone.
- Researchers can use the proposed heuristic as a baseline for adaptive or metaheuristic scheduling studies.
- CloudSim-based evaluation provides a controlled environment for testing policies before deployment on production infrastructure.
- The approach can be extended with network- and memory-aware placement to align more closely with multi-resource scheduling requirements.

9. Limitations and Threats to Validity

The source thesis acknowledges several limitations: the study is primarily focused on cloud task scheduling; the analysis relies on theoretical models and secondary data; real-time large-scale validation is limited; dynamic workload fluctuations may not be fully represented; advanced scheduling can introduce computational overhead; and cloud technologies evolve rapidly. filecite turn1file0 L53-L70

These limitations should be considered when interpreting the results. In particular, CloudSim experiments are simulations rather than production deployments. Results can vary with workload distributions, VM sizes, host heterogeneity, network assumptions, and simulator configuration. The thesis also notes that comparisons can be affected by differences in simulation environment or configuration. Filecite turn2file3 L235-L246

Accordingly, the reported findings should be interpreted as evidence of performance under the specified simulation conditions, not as a universal guarantee across all cloud platforms. A stronger journal version should report complete simulator parameters, random seeds where applicable, workload traces, number of replications, confidence intervals, and statistical significance tests when the raw experimental data are available.

10. Conclusion and Future Work

This paper presented an energy- and SLA-aware heuristic for cloud resource management based on host overload/underload detection, migration-time-aware VM selection, VM consolidation, and host shutdown. The thesis evidence demonstrates that conventional Min-Min scheduling is effective for reducing completion time in the baseline experiment, but that execution time is only one dimension of cloud scheduling. The proposed infrastructure-level policy extends the scheduling perspective by incorporating SLA behavior, VM migration, host shutdown, and energy considerations.

The reported results are encouraging. In the direct existing-versus-proposed comparison, overall SLA violation decreases from 38.20% to 27.50%, while the detailed experiment reports 0.16% SLA degradation due to migration and fewer host shutdowns than the comparison policies. The thesis also reports lower energy consumption for the proposed approach. However, the slight increase in VM migrations indicates a trade-off that should be explicitly optimized in future versions.

Future work should therefore investigate adaptive overload thresholds, predictive workload models, multi-resource placement using CPU, memory, and bandwidth, and multi-objective optimization of SLA, energy, migration cost, and makespan. Further evaluation using real workload traces and heterogeneous cloud configurations would strengthen external validity. Hybrid methods combining the proposed heuristic with metaheuristic or learning-based decision mechanisms may also provide a promising route toward robust dynamic scheduling.

References

1. Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., Lee, G., Patterson, D. A., Rabkin, A., & Zaharia, M. (2009). Above the clouds: A Berkeley view of cloud computing (Technical Report No. UCB/EECS-2009-28). University of California, Berkeley.
2. Beloglazov, A., Abawajy, J., & Buyya, R. (2012). Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing. Future Generation Computer Systems, 28(5), 755–768.
3. Buyya, R., Ranjan, R., & Calheiros, R. N. (2009). Modeling and simulation of scalable Cloud Computing environments and the CloudSim toolkit: Challenges and opportunities. In Proceedings of the 7th High Performance Computing and Simulation Conference (pp. 1–11).
4. Calheiros, R. N., Ranjan, R., Beloglazov, A., De Rose, C. A. F., & Buyya, R. (2011). CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. Software: Practice and Experience, 41(1), 23–50.
5. Akilandeswari, P., & Srimathi, H. (2016). Survey and analysis on task scheduling in cloud environment. Indian Journal of Science and Technology, 9, 1–6.
6. Annette, R., Banu, A., & Shriram, S. (2013). A taxonomy and survey of scheduling algorithms in cloud based on task dependency. International Journal of Computer Applications, 82, 20–26.
7. Lee, Y. C., & Zomaya, A. Y. (2012). Energy efficient utilization of resources in cloud computing systems. The

Journal of Supercomputing, 60, 268–280.
8. Malekloo, A., & collaborators. (2018). QoS-aware and energy-efficient VM consolidation and placement using optimization approaches. [Bibliographic details should be verified against the original source before submission].
9. Yadav, R., & collaborators. (2018). Adaptive approaches for reducing SLA violations and energy consumption in cloud environments. [Bibliographic details should be verified against the original source before submission].
10. Topcuoglu, H., Hariri, S., & Wu, M. (2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Transactions on Parallel and Distributed Systems, 13(3), 260–274.