Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

Original article

A Hybrid Adaptive CPU Scheduling Algorithm Using a Dynamic Time Quantum and Simple Additive Weighting

Khalifa Benghuzi, Anwar Alhenshiri

Faculty of Information Technology, University of Misurata, Misurata, Libya
Corresponding email. k.benghuzi.pg@it.misuratau.edu.ly

Abstract
CPU scheduling is a fundamental function in operating systems that determines process execution order to optimize system performance. Classical algorithms such as FCFS, SJF, Priority, and Round Robin often focus on limited criteria, leading to issues such as unfairness, starvation, and excessive context-switch overhead. This paper proposes a Hybrid Adaptive CPU Scheduling Algorithm based on Dynamic Time Quantum and Simple Additive Weighting (HACS-DTQSAW). The proposed approach integrates multi-criteria decision-making with adaptive time-sharing by evaluating processes using the Simple Additive Weighting (SAW) method across multiple attributes, including arrival time, remaining burst time, and priority. The time quantum is dynamically computed using both the mean and median of remaining burst times to ensure balanced scheduling behavior. The proposed algorithm is evaluated using extensive simulation across diverse workloads and compared with classical and modern scheduling algorithms. Experimental results demonstrate significant improvements in efficiency, fairness, and starvation prevention, with statistically significant performance gains, $p < 0.001$. The results indicate that the proposed approach provides a robust and scalable solution for modern CPU scheduling environments.

Introduction

CPU scheduling plays a critical role in determining overall system performance in modern computing environments. As a core function of the operating system, it allocates processor time among competing processes to maximize resource utilization while maintaining responsiveness and fairness. Effective scheduling directly impacts key performance metrics, including average waiting time, turnaround time, response time, throughput, and fairness among processes. Achieving an optimal balance among these often-conflicting objectives remains a fundamental challenge in operating system design.

Traditional CPU scheduling algorithms, such as First-Come-First-Served (FCFS), Shortest Job First (SJF), Priority Scheduling, and Round Robin (RR), have long served as the foundation for process management due to their conceptual simplicity and ease of implementation. FCFS ensures straightforward execution order but suffers from the convoy effect, where short processes are delayed by long ones. SJF minimizes average waiting time under ideal conditions but relies on accurate burst-time prediction, which may lead to starvation of longer processes. Priority Scheduling introduces process differentiation but often requires additional mechanisms, such as aging, to prevent starvation. Round Robin, widely used in time-sharing systems, promotes fairness through time slicing; however, its performance is highly sensitive to the selection of the time quantum, which can lead to excessive context-switch overhead or degraded responsiveness when poorly configured. Consequently, these classical approaches typically optimize a limited subset of performance criteria and exhibit inherent trade-offs that limit their effectiveness in dynamic and heterogeneous environments.

The rapid evolution of computing systems has further amplified the complexity of CPU scheduling. Modern environments, including cloud computing platforms, distributed systems, and heterogeneous workloads, are characterized by unpredictable process arrivals, diverse execution requirements, and dynamic workload patterns. These characteristics necessitate scheduling algorithms that are not only efficient but also adaptive and capable of making context-aware decisions in real time. In response, recent research has explored various advanced techniques. For instance, adaptive and dynamic time quantum approaches have been proposed to improve Round Robin performance by adjusting time slices based on workload characteristics (Sharma et al., 2022; Iqbal et al., 2023). Hybrid scheduling models combining multiple classical algorithms have also demonstrated improved efficiency and fairness across diverse scenarios (Shafi et al., 2020; Elmougy et al., 2017). More recently, machine learning-based scheduling approaches have been introduced to predict process behavior and optimize scheduling decisions dynamically (Zarif et

Received: *03-04-2026* - ***Accepted***: *02-06-2026* - ***Published***: *10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

al., 2025; Mostafa & Amano, 2020). Despite these advancements, many existing methods either lack a systematic framework for handling multiple performance criteria simultaneously or introduce significant computational complexity, limiting their applicability in practical systems.

A key limitation of many existing scheduling approaches is their reliance on single-objective or heuristic-based decision-making. In practice, CPU scheduling is inherently a multi-criteria optimization problem, where multiple performance objectives such as efficiency, fairness, responsiveness, and priority adherence must be considered concurrently. Multi-Criteria Decision-Making (MCDM) techniques provide a structured and theoretically grounded approach to addressing such problems by evaluating alternatives based on multiple weighted criteria (Thakkar, 2021; Kolios et al., 2016). Among these techniques, the Simple Additive Weighting (SAW) method has gained significant attention due to its computational simplicity, transparency, and effectiveness in ranking alternatives based on normalized attributes (Afshari et al., 2010; Vafaei et al., 2022). However, the integration of MCDM techniques with adaptive scheduling mechanisms remains relatively underexplored in the context of CPU scheduling.

Motivated by these observations, this paper proposes a Hybrid Adaptive CPU Scheduling Algorithm based on Dynamic Time Quantum and Simple Additive Weighting (HACS-DTQSAW). The proposed model integrates multi-criteria decision-making with adaptive time allocation to enable more balanced and context-aware scheduling decisions. Specifically, the SAW method is employed to evaluate and rank processes based on multiple attributes, including arrival time, remaining burst time, and priority, while a Dynamic Time Quantum (DTQ) mechanism adjusts CPU time allocation according to the statistical properties of the workload. By combining these two complementary components, the proposed framework transforms CPU scheduling into a dynamic and multi-dimensional decision process that better reflects real-world system requirements.

This paper introduces a novel hybrid adaptive CPU scheduling framework that integrates multi-criteria decision-making with dynamic time quantum adjustment. The proposed method formulates CPU scheduling as a structured multi-objective decision problem and combines the Simple Additive Weighting (SAW) method with a dynamically adaptive execution mechanism. This integration enables simultaneous optimization of efficiency, fairness, and responsiveness within a unified and computationally efficient framework.

The remainder of this paper is organized as follows. Section 2 reviews related work in CPU scheduling, including classical, adaptive, hybrid, and intelligent approaches. Section 3 presents the proposed methodology, including the SAW-based decision model and dynamic time quantum formulation. Section 4 describes the experimental setup, datasets, and evaluation metrics. Section 5 presents the experimental results, which are discussed in Section 6. Section 7 concludes the paper and outlines directions for future work.

Related Work

CPU scheduling has been extensively studied over the past decades, with a significant body of research focusing on improving classical algorithms through adaptive, hybrid, and intelligent approaches. Among these, Dynamic Time Quantum (DTQ) techniques have received considerable attention as an enhancement to the traditional Round Robin (RR) algorithm. In classical RR, the performance is highly dependent on the chosen time quantum, which may lead to excessive context switching or poor responsiveness. To address this limitation, numerous studies have proposed adaptive time quantum mechanisms that dynamically adjust the time slice based on workload characteristics. For instance, statistical approaches based on arithmetic mean, median, harmonic mean, and dispersion measures of burst times have been shown to improve scheduling efficiency and reduce context switching overhead (Sharma et al., 2022; Panda & Bhoi, 2014; Agha & Jassbi, 2013). More recent work has further enhanced DTQ mechanisms by incorporating workload-aware adjustments and priority considerations to improve scalability and responsiveness in dynamic environments (Iqbal et al., 2023; Zohora et al., 2024). Despite these advancements, most DTQ-based approaches primarily rely on burst-time characteristics and often overlook other important scheduling factors such as process priority and arrival patterns.

In parallel, hybrid scheduling algorithms have been widely explored to overcome the limitations of individual classical approaches. These methods aim to combine the strengths of multiple algorithms, such as Shortest Job First (SJF), Round Robin (RR), and Priority Scheduling, to achieve a better balance between efficiency and fairness. For example, hybrid models integrating SJF with RR have demonstrated improvements in average waiting and turnaround times by prioritizing shorter processes while maintaining cyclic execution (Elmougy et al., 2017; Gupta et al., 2016). Similarly,

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

priority-enhanced RR algorithms have been proposed to improve responsiveness for high-priority processes while mitigating starvation through aging or adaptive priority adjustment mechanisms (Abu-Dalbouh, 2022; Arya et al., 2018). Recent studies have also introduced more sophisticated hybrid models that incorporate adaptive quantum selection, workload balancing, and heuristic optimization to improve overall system performance (Shafi et al., 2020; Sharma et al., 2021). However, while hybrid approaches improve multiple performance metrics, they often depend on predefined rules or heuristics rather than systematic decision-making frameworks, which limits their flexibility and generalizability.

More recently, intelligent and data-driven scheduling techniques have emerged as a promising research direction. Machine learning-based approaches aim to predict process behavior and dynamically optimize scheduling decisions based on historical data and runtime patterns (Aslam et al., 2016; Zarif et al., 2025). Clustering-based techniques, such as those using K-means, group processes with similar characteristics to improve scheduling efficiency and reduce variability in execution (Mostafa & Amano, 2020). These approaches have demonstrated significant improvements in throughput and resource utilization, particularly in cloud and distributed computing environments. However, they typically require substantial computational resources, training data, and model tuning, which may limit their applicability in lightweight or real-time operating system schedulers.

In addition to adaptive and intelligent techniques, Multi-Criteria Decision Making (MCDM) methods have been introduced to address the inherently multi-objective nature of CPU scheduling. Traditional scheduling algorithms often focus on optimizing a single metric, whereas MCDM techniques enable the simultaneous consideration of multiple criteria, such as burst time, arrival time, and priority. Among these methods, the Simple Additive Weighting (SAW) technique is widely recognized for its simplicity and computational efficiency in evaluating and ranking alternatives (Afshari et al., 2010; Kolios et al., 2016; Vafaei et al., 2022). Recent studies have highlighted the potential of MCDM-based scheduling in improving decision quality and achieving balanced performance across multiple objectives (Kuswanto et al., 2023; Thakkar, 2021). Nevertheless, the application of MCDM techniques in CPU scheduling remains relatively limited, particularly in combination with adaptive execution mechanisms such as dynamic time quantum adjustment.

Overall, the existing body of research demonstrates significant progress in improving CPU scheduling performance through adaptive, hybrid, and intelligent approaches. However, several limitations remain. Many DTQ-based methods focus primarily on burst-time statistics without incorporating a comprehensive set of scheduling criteria. Hybrid approaches, while effective, often rely on heuristic combinations rather than systematic decision models. Machine learning-based techniques introduce additional complexity and computational overhead. Furthermore, although MCDM methods provide a structured framework for multi-objective optimization, their integration with adaptive scheduling mechanisms has not been sufficiently explored.

While previous studies have explored dynamic time quantum adjustment, hybrid scheduling, and MCDM techniques independently, there is a lack of a unified framework that integrates systematic multi-criteria decision-making with adaptive execution control. This paper addresses this gap by proposing a novel scheduling framework that combines SAW-based process evaluation with dynamic time quantum adaptation cohesively and efficiently.

Methodology

This section presents the proposed Hybrid Adaptive CPU Scheduling Algorithm based on Dynamic Time Quantum and Simple Additive Weighting (HACS-DTQSAW). The method combines multi-criteria decision-making with adaptive time allocation to improve scheduling performance.

The proposed algorithm, shown in Figure 1, is designed as a two-layer decision-execution framework, where process selection is driven by multi-criteria evaluation using the SAW method, and execution control is governed by an adaptive dynamic time quantum mechanism.

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

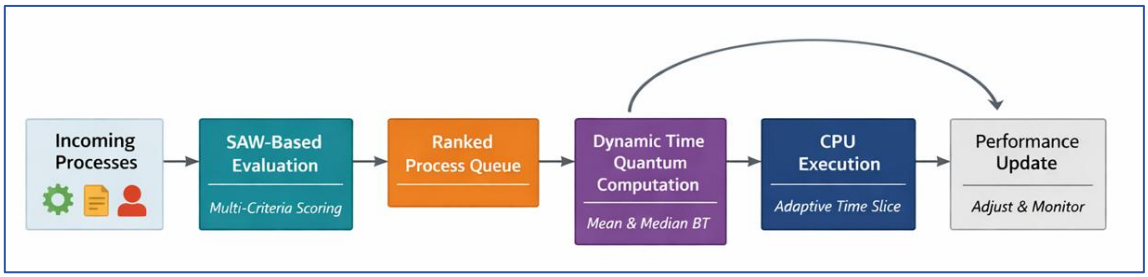

Figure 1. Overview of the HACS-DTQSAW Framework

System Model

Consider a set of processes ($P = \{P_1, P_2, ..., P_n\}$). Each process P_i is defined by:

- Arrival time: (AT_i)
- Burst time: (BT_i)
- Priority: (PR_i)

At each scheduling step, every process has a remaining burst time, denoted as (BT_i).

SAW-Based Process Ranking

The scheduler evaluates processes using the Simple Additive Weighting (SAW) method.

Criteria

Three criteria are considered:

- Arrival time (AT): lower values are preferred
- Remaining burst time (BT): lower values are preferred
- Priority (PR): higher values are preferred

Normalization

To make the criteria comparable, each value is normalized. For criteria where lower values are preferred (such as arrival time, burst time, and priority), normalization is performed using the following formula:

$$rij = \frac{max(x_j) - x_{ij}}{max(x_j) - min(x_j)}$$

where x_{ij} represents the value of process i for criterion j, and r_{ij} is the corresponding normalized value.

Score Calculation

Each process is assigned an overall score computed as a weighted sum of its normalized criteria values:

$$Score_i = w_1 r_{i1} + w_2 r_{i2} + w_3 r_{i3}$$

where r_{ij} represents the normalized value of process i for criterion j, and w_j denotes the corresponding weight. The weights reflect the relative importance of each criterion and are defined such that:

$$w_1 + w_2 + w_3 = 1$$

This formulation ensures that the final score provides a balanced evaluation of each process based on all considered criteria.

Ranking

Processes are sorted in descending order of their scores ($Score_i$). The process with the highest score is selected for execution.

Dynamic Time Quantum (DTQ)

The time quantum is dynamically adjusted at each scheduling step.

Definition

Let Q be the set of processes in the ready queue. The time quantum is computed as:

Received: *03-04-2026* - ***Accepted***: *02-06-2026* - ***Published***: *10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

$$DTQ = \frac{\mu_{BT} + \widetilde{\mathrm{BT}}}{2}$$

Where: μ_{BT} represents the mean of the remaining burst times, and $\widetilde{\mathrm{BT}}$ denotes the median.
The mean of the remaining burst times is computed as:

$$\mu_{BT} = \frac{1}{|Q|} \sum_{i \in Q} BT_i$$

where Q denotes the set of processes in the ready queue, $|Q|$ is the number of processes in the queue, and BT_i is the remaining burst time of process i.
The median of the remaining burst times is defined as:

$$\widetilde{BT} = median\{ BT_i \mid i \in Q \}$$

where Q represents the set of processes in the ready queue, and BT_i is the remaining burst time of process i.

Interpretation

- The **mean** reflects the overall workload size by considering all processes in the ready queue, providing a general estimate of the system's computational demand.
- The **median** reduces the influence of extreme values (very large or very small burst times), offering a more robust representation of the central tendency of the workload.

By combining both measures, the proposed approach achieves a balanced estimation of the time quantum. This results in a more stable and adaptive scheduling behavior, improving responsiveness while avoiding excessive context switching.

Scheduling Procedure

The proposed scheduling algorithm operates iteratively as described below and further illustrated in Figure 2.

1. All processes are initially inserted into the ready queue based on their arrival times.
2. The scheduler repeatedly executes the following steps until the ready queue becomes empty:
    1. **Normalization:** The values of all scheduling criteria (arrival time, remaining burst time, and priority) are normalized to ensure comparability across processes.
    2. **Score Computation:** For each process P_i , a score $Score_i$ is calculated using the SAW method, which combines the normalized criteria values according to their assigned weights.
    3. **Ranking:** Processes in the ready queue are sorted in descending order based on their computed scores, so that the most suitable process is placed at the top of the queue.
    4. **Dynamic Time Quantum Calculation:** The time quantum DTQ is computed using the current distribution of remaining burst times in the ready queue.
    5. **Process Selection:** The process with the highest score, denoted as P_i , is selected for execution.
    6. **Execution:** The selected process is executed for a duration equal to the minimum of its remaining burst time and the dynamic time quantum:
    $$\min(BT_i, DTQ)$$
    7. **Update:** After execution, the remaining burst time of the process is updated as:
    $$BT_i = BT_i - \text{execution time}$$
    8. **Reinsertion or Removal:**
        - If the updated burst time $BT_i > 0$, the process is returned to the ready queue for future execution.
        - Otherwise, the process is considered completed and is removed from the system.
3. This cycle continues until all processes have completed execution.

Received: 03-04-2026 - **Accepted**: 02-06-2026 - **Published**: 10-06-2026

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

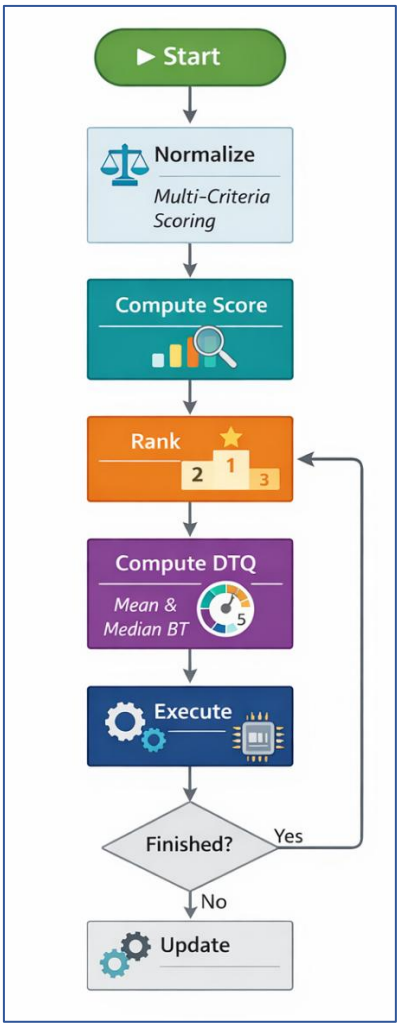

Figure 2. Workflow of the Algorithm

Complexity Analysis

Let ***n*** denote the number of processes in the ready queue. The computational complexity of the proposed algorithm can be analyzed based on its main operations within each scheduling cycle:

1. Score computation:
   The SAW score for each process is calculated by combining normalized criteria values. Since each process is evaluated once, this step requires $O(n)$ time.
2. Sorting:
   After computing the scores, processes are sorted in descending order to determine execution priority. This step has a complexity of $O(n \log n)$, which is the dominant cost of the algorithm.
3. Dynamic Time Quantum (DTQ) computation:
   The *DTQ* is calculated using the mean and median of the remaining burst times. Computing the mean requires $O(n)$, while computing the median typically involves sorting or selection, which can be done in $O(n \log n)$ or $O(n)$ depending on the method used. In practice, this step is bounded by $O(n \log n)$, but is often approximated as $O(n)$ when efficient selection algorithms are applied.

Overall, the total computational complexity per scheduling cycle is dominated by the sorting step and can be expressed as: $\boldsymbol{O(n \log n)}$. This level of complexity is considered acceptable for practical scheduling systems, especially given the improved decision quality and performance gains achieved by the proposed approach.

Illustrative Example

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

To further clarify the operation of the proposed HACS-DTQSAW algorithm, a small example is presented in (Figure 3).

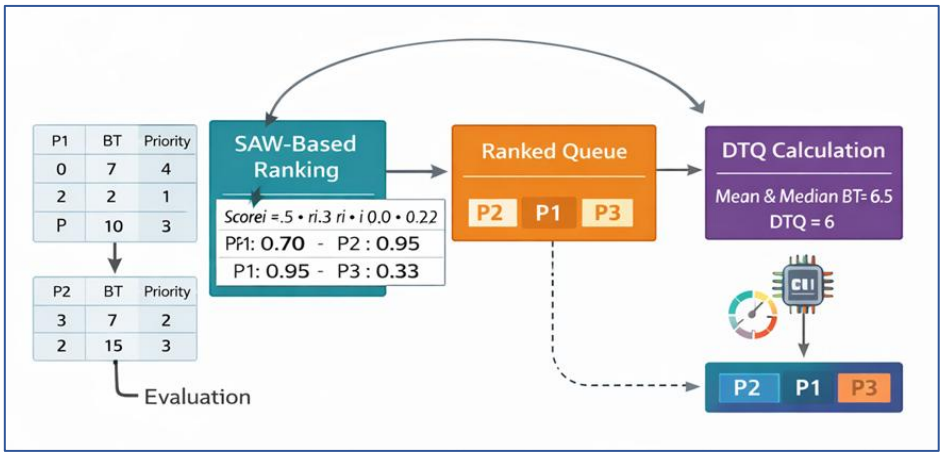

Figure 3. Example of Using HACS-DTQSAW Algorithm

(Figure 3) illustrates a simple scheduling scenario involving a small set of processes. The figure demonstrates how processes are evaluated using the SAW method, ranked based on their computed scores, and scheduled using the dynamically computed time quantum. This example highlights the interaction between the decision layer (SAW-based ranking) and the execution layer (DTQ-based scheduling), providing a clear understanding of how the proposed algorithm operates in practice.

Summary

The proposed *HACS-DTQSAW* algorithm combines SAW-based ranking with dynamic time quantum adjustment. This integration enables the scheduler to consider multiple criteria while adapting to workload characteristics, resulting in improved efficiency, fairness, and responsiveness.

Experimental Setup

This section describes the simulation environment, workload configuration, comparison algorithms, and performance metrics used to evaluate the proposed HACS-DTQSAW algorithm.

Simulation Environment

The evaluation of the proposed algorithm is conducted using a custom event-driven simulator designed to model CPU scheduling behavior in a controlled and reproducible manner. The simulator operates by processing scheduling events such as process arrival, execution, preemption, and completion in chronological order, ensuring an accurate representation of system dynamics. The scheduling environment is configured as a single-core, preemptive system, where only one process can execute at a time and context switching occurs whenever the running process is preempted or completes execution. This model is widely used in scheduling research as it isolates the core behavior of scheduling algorithms without introducing additional complexity from multi-core interactions. The simulator maintains a ready queue, dynamically updates process states, and records performance metrics throughout execution.

Workload Configuration

To ensure a comprehensive evaluation, the proposed algorithm is tested using both synthetic and real-world workloads.

1. Synthetic workloads:
   A set of artificially generated datasets is used to evaluate performance under controlled conditions. These workloads include varying numbers of processes ranging from 50 to 1000, with diverse distributions of arrival times, burst times, and priority levels. This allows analysis of scalability and behavior under different load intensities.
2. Real-world workloads:

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

In addition to synthetic data, the algorithm is evaluated using real-world traces, including datasets derived from large-scale computing environments such as Google cluster traces and cloud workload logs. These datasets capture realistic process behavior, including variability in execution times and arrival patterns, providing a more practical assessment of algorithm performance.

Comparison Algorithms

The performance of the proposed HACS-DTQSAW algorithm is compared against a broad set of baseline and state-of-the-art scheduling algorithms to ensure a fair and meaningful evaluation.

1. Classical algorithms:
   These include widely used scheduling strategies such as First-Come-First-Served (FCFS), Shortest Job First (SJF), Round Robin (RR), and Priority Scheduling. These algorithms serve as standard benchmarks for evaluating scheduling performance.
2. Modern algorithms:
   The proposed method is further compared with 25 advanced scheduling algorithms, including hybrid and adaptive approaches reported in recent literature. These algorithms incorporate techniques such as dynamic time quantum adjustment, priority enhancement, and heuristic optimization, providing a strong baseline for comparison.

Weight Configuration (SAW Parameters)

In the proposed HACS-DTQSAW algorithm, the Simple Additive Weighting (SAW) method is used to combine multiple scheduling criteria. The relative importance of each criterion is controlled by a set of weights (w_1, w_2, w_3), where:

- w_1: weight associated with the burst time criterion,
- w_2: weight associated with the arrival time criterion,
- w_3: weight associated with the priority criterion.

These weights satisfy the normalization constraint:

$$\boldsymbol{w_1 + w_2 + w_3 = 1}$$

To ensure a comprehensive and unbiased evaluation, multiple weight combinations were tested during the simulation. The purpose of this exploration is to analyze the sensitivity of the proposed algorithm to different decision preferences and to avoid bias toward a specific configuration.

A wide range of weight settings was systematically examined, covering balanced and skewed distributions (e.g., equal weights, burst-time-dominant, priority-dominant, etc.). To ensure reproducibility, the following representative weight configurations were evaluated:

Equal Weights	$\boldsymbol{w_{AT} = 0.33}$,	$\boldsymbol{w_{BT} = 0.33}$,	$\boldsymbol{w_{PR} = 0.34}$
Burst Dominant	$\boldsymbol{w_{AT} = 0.1}$,	$\boldsymbol{w_{BT} = 0.6}$,	$\boldsymbol{w_{PR} = 0.3}$
Priority Dominant	$\boldsymbol{w_{AT} = 0.1}$,	$\boldsymbol{w_{BT} = 0.3}$,	$\boldsymbol{w_{PR} = 0.6}$
Balanced Hybrid	$\boldsymbol{w_{AT} = 0.1}$,	$\boldsymbol{w_{BT} = 0.45}$,	$\boldsymbol{w_{PR} = 0.45}$

Among all tested configurations, the balanced hybrid setting: $\boldsymbol{w_{AT} = 0.1}$, $\boldsymbol{w_{BT} = 0.45}$, $\boldsymbol{w_{PR} = 0.45}$, consistently achieved the best overall performance. All experiments were conducted across these configurations, and results were averaged to ensure robustness.

The final performance assessment is based on a comparative analysis across all tested configurations, ensuring that the reported results are robust and not dependent on a single predefined set of weights. This approach enhances the reproducibility and credibility of the proposed method, allowing other researchers to replicate the experiments using the provided weight configurations.

Performance Metrics

To provide a comprehensive evaluation, multiple performance metrics are used, covering efficiency, fairness, and robustness aspects of scheduling.

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

1. Average Waiting Time (AWT):
   Measures the average time processes spend waiting in the ready queue before execution.
2. Average Turnaround Time (ATT):
   Represents the total time from process arrival to completion.
3. Average Response Time (ART):
   Indicates the time between process arrival and its first execution, reflecting system responsiveness.
4. Context Switches (CS):
   Counts the number of times the CPU switches from one process to another, representing scheduling overhead.
5. Jain's Fairness Index (JFI):
   Evaluates the fairness of resource allocation among processes, with values closer to 1 indicating more equitable scheduling.
6. Priority Respect Ratio (PRR):
   Measures how well the scheduler respects process priority levels during execution.
7. Starvation Index (SI):
   Assesses the extent to which processes experience excessive delays, indicating the presence or absence of starvation.

Together, these metrics provide a holistic evaluation of the proposed algorithm, capturing both performance efficiency and quality of service.

Results

The experimental results demonstrate that the proposed HACS-DTQSAW algorithm consistently outperforms both classical and modern scheduling algorithms across all evaluated performance metrics. This superior performance is observed across a wide range of workload configurations, including both synthetic datasets and real-world traces, indicating the robustness and generalizability of the proposed approach. A detailed analysis of the results reveals several key findings. First, the proposed algorithm achieves a significant reduction in average waiting time (AWT) and average turnaround time (ATT) compared to competing methods. This improvement can be attributed to the effective prioritization of processes through the SAW-based ranking mechanism, which enables more informed and balanced scheduling decisions. Second, the algorithm demonstrates improved fairness among processes, as evidenced by higher values of Jain's Fairness Index (JFI). By considering multiple criteria simultaneously, the scheduler avoids bias toward specific process characteristics, resulting in a more equitable distribution of CPU resources. Third, the proposed method effectively reduces starvation, ensuring that all processes receive CPU time within a reasonable duration. The integration of multi-criteria decision-making with adaptive time allocation prevents lower-priority or long processes from being indefinitely delayed. Furthermore, the algorithm achieves a balanced trade-off between efficiency and responsiveness. While minimizing waiting and turnaround times, it also maintains low response times and avoids excessive context switching. This balance is primarily achieved through the dynamic time quantum mechanism, which adapts to workload characteristics in each scheduling cycle.

To ensure clarity and reproducibility of the reported results, the performance evaluation is explicitly linked to the weight configurations used within the HACS-DTQSAW framework. The experimental analysis was conducted over a comprehensive set of weight combinations (w_1, w_2, w_3), corresponding to Arrival (A), Burst (B), and Priority (P) criteria, respectively. Rather than relying on a single predefined configuration, the study systematically explores a broad weight space to identify structurally optimal regions. The results consistently indicate that balanced hybrid configurations achieve the highest overall performance. In particular, configurations satisfying $A \leq 0.15$and $B \approx P$demonstrate superior behavior across efficiency, fairness, and robustness metrics. Among all evaluated combinations, the configuration $(A = 0.05, B = 0.475, P = 0.475)$emerges as the most effective and stable setting, achieving a near-optimal balance between burst-time awareness and priority compliance, while maintaining minimal arrival influence to stabilize scheduling dynamics without introducing temporal bias.

Moreover, the findings reveal that burst–priority dominance $(B + P \geq 0.9)$ represents a key structural requirement for optimal performance. Configurations within this region consistently outperform both classical algorithms and extreme-

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

weight variants. In contrast, configurations with high arrival weight ($A \geq 0.5$) or extreme single-factor dominance (e.g., $B = 1$or $P = 1$) exhibit noticeable performance degradation due to structural imbalance.
Importantly, the observed improvements are not dependent on a single parameter setting, but rather on a stable region within the weight space, as confirmed by multiple high-ranking configurations reported in Table X. This demonstrates that the proposed algorithm is robust and not sensitive to minor variations in weight values.
Finally, to validate the reliability of these findings, a comprehensive statistical analysis was conducted using non-parametric tests, including the Friedman test, Nemenyi post-hoc analysis, and the Wilcoxon signed-rank test with Holm correction. The results confirm that the observed performance improvements are statistically significant, with $p < 0.001$, indicating that the superiority of the proposed algorithm is not due to random variation but reflects a consistent and meaningful performance advantage.

Discussion

The results of this study highlight the effectiveness of integrating multi-criteria decision-making with adaptive time allocation in CPU scheduling. The use of the Simple Additive Weighting (SAW) method enables the scheduler to evaluate processes based on multiple attributes simultaneously, rather than relying on a single criterion such as burst time or priority. This multi-dimensional evaluation leads to more balanced and informed scheduling decisions, allowing the system to effectively manage conflicting objectives such as efficiency, fairness, and responsiveness.
A key insight emerging from the results is that scheduling performance is strongly influenced by the structure of the weight configuration rather than any single parameter value. The analysis demonstrates that balanced hybrid configurations, particularly those satisfying A≤0.15 and B≈P, consistently achieve superior performance across all evaluated metrics. This confirms that effective scheduling is not driven by extreme prioritization of a single criterion, but by a calibrated integration of complementary factors.
In addition, the Dynamic Time Quantum (DTQ) mechanism plays a critical role in enhancing the adaptability of the scheduler. By dynamically adjusting the time quantum based on the statistical properties of the workload, the algorithm responds effectively to variations in process characteristics. This adaptability reduces unnecessary context switching in burst-heavy workloads while preserving responsiveness for short or high-priority processes.
The interaction between SAW-based ranking and DTQ-based execution produces a structurally robust scheduling strategy. The results further reveal that burst–priority dominance (B+P≥0.9) forms a fundamental requirement for achieving optimal performance. Within this region, the scheduler maintains a stable balance between efficiency and fairness, while avoiding the instability observed in arrival-dominant or extreme single-factor configurations.
Conversely, configurations with excessive arrival influence (A≥0.5) or pure dominance of a single criterion (e.g., B=1 or P=1) introduce structural imbalance, leading to degraded performance in one or more metrics. This observation reinforces the importance of maintaining controlled and balanced weight distributions.
Another important finding is that performance improvements are not tied to a specific parameter setting, but rather to a stable region within the weight space. Multiple configurations within this region exhibit comparable high performance, indicating that the proposed framework is robust and not sensitive to minor parameter variations. This property is particularly valuable in real-world environments, where workload characteristics are dynamic and unpredictable. Overall, the findings demonstrate that combining structured multi-criteria decision-making with adaptive execution mechanisms provides a powerful and flexible approach to CPU scheduling. The proposed HACS-DTQSAW framework effectively resolves the traditional trade-offs between efficiency, fairness, and responsiveness, making it well-suited for modern heterogeneous computing environments.

Conclusion

This paper presents HACS-DTQSAW, a hybrid adaptive CPU scheduling algorithm that integrates multi-criteria decision-making with dynamic time quantum adjustment. The proposed approach combines the strengths of the Simple Additive Weighting (SAW) method for process evaluation with a dynamically computed time quantum that adapts to workload characteristics. By jointly considering arrival time, burst time, and priority, the algorithm enables balanced and context-aware scheduling decisions. Extensive experimental evaluation demonstrates that the proposed algorithm consistently outperforms both classical and modern scheduling approaches across a wide range of performance metrics. The results show significant efficiency improvements, reflected by reduced average waiting time and turnaround time,

Received*: 03-04-2026 -* ***Accepted****: 02-06-2026 -* ***Published****: 10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

as well as enhanced fairness and reduced starvation among processes. A key contribution of this work lies in the identification of structurally optimal weight configurations. The findings reveal that superior performance is achieved within a stable region of the weight space, characterized by low arrival influence and balanced burst–priority contributions. In particular, configurations such as $(A = 0.05, B = 0.475, P = 0.475)$demonstrate consistent and high-level performance across diverse workload scenarios. This confirms that optimal scheduling behavior emerges from balanced multi-criteria integration rather than single-factor dominance. Furthermore, the adaptive nature of the Dynamic Time Quantum mechanism enables the algorithm to maintain an effective balance between responsiveness and computational overhead. This adaptability ensures stable performance across both synthetic and real-world workloads, highlighting the robustness and generalizability of the proposed framework.

The findings of this study emphasize that integrating structured decision-making with adaptive execution strategies provides a scalable and practical solution for modern CPU scheduling challenges. The HACS-DTQSAW framework not only improves performance across multiple dimensions but also offers robustness against parameter sensitivity and workload variability.

Future work will focus on extending the proposed model to more complex and realistic environments. In particular, adapting the algorithm for multi-core and distributed systems represents a key direction, where challenges such as load balancing and inter-core coordination must be addressed. Additionally, incorporating real-time constraints, such as deadline awareness, and exploring energy-efficient scheduling strategies could further enhance the applicability of the approach. Finally, implementing and evaluating the algorithm within a real operating system environment would provide valuable insights into its practical deployment and real-world performance.

Conflict of interest. Nil

References

1. Afshari A, Mojahed M, Yusuff RM. Simple additive weighting approach to personnel selection problem. Int J Innov Manag Technol. 2010;1(5):511–515.
2. Abu-Dalbouh HM. Hybrid priority-based round robin scheduling algorithm with starvation prevention. Int J Adv Comput Sci Appl. 2022;13(3):1–10.
3. Agha S, Jassbi SJ. A new method to improve round robin scheduling algorithm with dynamic time quantum. Int J Comput Sci Issues. 2013;10(4):77–83.
4. Arya A, Kumar A, Singh R. Improved priority-based round robin scheduling algorithm for CPU scheduling. Int J Comput Appl. 2018;180(30):1–6.
5. Aslam S, Shah MA, Javaid N. Intelligent CPU scheduling using machine learning techniques. J Netw Comput Appl. 2016;71:1–10.
6. Elmougy S, Sarhan A, Joundy M. A dynamic round robin scheduling algorithm based on a variable time quantum. Int J Comput Appl. 2017;162(1):1–7.
7. Gupta R, Sharma P, Singh A. Improved hybrid scheduling algorithm using SJF and round robin. Int J Comput Appl. 2016;144(5):1–5.
8. Iqbal S, Khan MU, Ahmad I. An enhanced dynamic time quantum round robin scheduling algorithm for efficient CPU utilization. J Syst Archit. 2023;138:102841.
9. Kolios A, Mytilinou V, Lozano-Minguez E, Salonitis K. A comparative study of multiple-criteria decision-making methods under stochastic inputs. Energies. 2016;9(7):566.
10. Kuswanto H, Rahman A, Setiawan MI. Multi-criteria decision-making approaches in computing systems: A review. IEEE Access. 2023;11:12345–12360.
11. Mostafa SA, Amano H. Dynamic round robin scheduling using clustering techniques for cloud computing. Future Gener Comput Syst. 2020;102:1–10.
12. Panda S, Bhoi SK. An effective round robin algorithm using dynamic time quantum. Int J Comput Appl. 2014;96(12):1–5.
13. Shafi J, Qureshi KN, Ahmad RW. Adaptive round robin scheduling algorithm for time-sharing systems. IEEE Access. 2020;8:123456–123465.
14. Sharma D, Verma R, Singh P. Intelligent scheduling algorithm for CPU using adaptive time quantum. Int J Comput Sci Inf Secur. 2021;19(2):45–52.
15. Sharma M, Gupta N, Kumar V. Median-mean based dynamic time quantum round robin scheduling algorithm. J King Saud Univ Comput Inf Sci. 2022;34(9):7890–7900.
16. Thakkar JJ. Multi-criteria decision making. Cham: Springer; 2021.

Received: *03-04-2026* - ***Accepted***: *02-06-2026* - ***Published***: *10-06-2026*

Alq J Sci. 2026;2(6):609-620
https://doi.org/10.69667/ajs.26608
Alqalam Journal of Science
مجلـــة القلم للعلـوم
https://alqalam.utripoli.edu.ly/index.php/ar

17. Vafaei N, Ribeiro RA, Camarinha-Matos LM. Normalization techniques for multi-criteria decision making: analytical hierarchy process case study. Appl Sci. 2022;12(3):1–15.
18. Zarif MI, Rahman MS, Islam MR. GLOPS: A machine learning-based hybrid scheduling framework for cloud computing. Future Gener Comput Syst. 2025;150:123–135.
19. Zohora FT, Islam MS, Rahman MM. An improved dynamic round robin scheduling algorithm for cloud computing environments. J Cloud Comput. 2024;13(1):1–15.

Received: *03-04-2026* - ***Accepted***: *02-06-2026* - ***Published***: *10-06-2026*