Advance Sustainable Science, Engineering and Technology (ASSET)
Vol. 8, No.3, July 2026, pp. 02603029-01 ~ 02603029-011
ISSN: 2715-4211 DOI: https://doi.org/10.26877/asset.v8i3.3122

# Scalable TOPSIS Variants in Web-Based Decision Support Systems: A Performance Benchmarking Study on Vectorization and Uncertainty Modelling

**Ghufron Abdullah[1*], Nugroho Dwi Saputro[2], Nurkolis[1]**

[1]Faculty of Post Graduate, Universitas Persatuan Guru Republik Indonesia Semarang, Jl. Sidodadi-Timur No.24 Semarang, Central Java 50232, Indonesia

[2]Faculty of Engineering and Informatics, Universitas Persatuan Guru Republik Indonesia Semarang, Jl. Sidodadi Timur No.24, Semarang, Central Java 50232, Indonesia.

*ghufronabdullah@upgris.ac.id

**Abstract**. This study systematically benchmarks the computational performance and decision consistency of three TOPSIS variants—classical TOPSIS, vectorized TOPSIS, and Z-TOPSIS—in scalable web-based multi-criteria decision support systems (DSS). Although TOPSIS is widely used due to its simplicity and interpretability, its scalability and computational behavior under concurrent web workloads remain insufficiently explored. To address this gap, the algorithms were evaluated using datasets containing 50–500 alternatives and 10 criteria, representing realistic decision-making scenarios. Classical TOPSIS was used as the baseline, vectorized TOPSIS applied matrix-based optimization, and Z-TOPSIS incorporated Z-numbers to capture uncertainty. Experiments were conducted on a cloud-based DSS equipped with multi-core CPUs and 16–32 GB RAM, measuring execution time, throughput, response time, and ranking consistency under workloads of 50–500 concurrent users. The results show that vectorized TOPSIS reduced execution time by approximately 50–55% (26.3 ms vs. 57.9 ms) and achieved the highest throughput of 480 requests per second. In contrast, Z-TOPSIS produced higher latency (68.4 ms) due to additional reliability computations. Ranking consistency remained high between classical and vectorized TOPSIS (Kendall's tau $\geq$ 0.98), while Z-TOPSIS showed minor deviations (tau = 0.91). These findings provide practical guidance for selecting TOPSIS variants in scalable web-based MCDM applications.

**Keywords**: Vectorized TOPSIS, Z-TOPSIS, scalable DSS, computational performance, MCDM benchmarking

***(Received 2025-12-09, Revised 2026-03-12, Accepted 2026-06-02, Available Online by 2026-07-01)***

## 1. Introduction

Multi-criteria decision-making (MCDM) methods play an important role in modern decision support systems (DSS), especially in situations where decision makers must evaluate multiple alternatives based on several criteria simultaneously [1]. Among the various MCDM techniques, the Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) remains one of the most frequently applied methods because it provides a straightforward ranking mechanism while maintaining relatively low computational complexity [2], [3]. As a result, TOPSIS has been successfully implemented in many domains, including supply chain management, infrastructure planning, and software requirement evaluation [4]–[6], [28].

The increasing adoption of cloud-based and web-based DSS platforms has introduced new operational challenges that are not fully addressed by conventional TOPSIS implementations. Modern systems are expected to process larger decision matrices while simultaneously responding to a growing number of user requests in real time [7], [8]. This issue is particularly relevant in public-sector applications such as RKAS (Rencana Kegiatan dan Anggaran Sekolah), where a centralized platform must evaluate numerous budget proposals submitted concurrently by different stakeholders. Under these conditions, the sequential nature of classical TOPSIS calculations can become a performance bottleneck, affecting both system responsiveness and scalability [7].

### *1.1. Related Work and Research Gap*

Several studies have attempted to improve the computational performance of TOPSIS by introducing vectorized and parallel processing techniques. By transforming iterative calculations into matrix-based operations, these approaches can reduce processing overhead and improve execution efficiency on modern hardware architectures [7], [15], [16]. Other researchers have focused on enhancing decision quality through uncertainty-aware extensions, particularly Z-TOPSIS, which incorporates Z-number theory to represent both the uncertainty of information and the reliability of its source [9], [18]–[20]. Such extensions are especially useful when decision data are incomplete, ambiguous, or obtained from sources with varying levels of confidence.

Despite these advances, most existing studies evaluate TOPSIS variants in controlled computational environments and primarily emphasize algorithmic accuracy or execution speed [3], [5], [7]. Comparatively less attention has been given to how these methods behave when deployed within real-world web-based DSS architectures that must handle asynchronous processing and concurrent user access [8], [10], [14]. In particular, there is limited empirical evidence regarding the trade-off between computational efficiency and uncertainty modelling when classical TOPSIS, vectorized TOPSIS, and Z-TOPSIS are executed under the same operational conditions [3], [5], [22].

TOPSIS also remains attractive when compared with other popular MCDM methods such as AHP, VIKOR, and WASPAS because it offers a balanced combination of implementation simplicity, computational efficiency, and flexibility for large-scale decision environments [6], [9], [21], [22], [29]. However, previous studies rarely investigate whether the benefits of vectorization and uncertainty modelling can be maintained when the algorithms are subjected to realistic web workloads and high levels of concurrency [10], [23].

Motivated by these limitations, this study evaluates the performance of classical TOPSIS, vectorized TOPSIS, and Z-TOPSIS within a scalable web-based DSS environment. The evaluation focuses not only on computational efficiency but also on throughput, response time, and ranking consistency under workloads of up to 500 concurrent users. By providing a comprehensive benchmarking analysis, this study aims to offer practical guidance for selecting the most appropriate TOPSIS variant based on application requirements related to scalability, responsiveness, and decision robustness.

## 2. Methods

The benchmarking methodology was designed to evaluate the performance of classical TOPSIS, vectorized TOPSIS, and Z-TOPSIS in a web-based DSS. The experimental framework consisted of benchmarking protocols, algorithm implementations, and system specifications to ensure consistent and reproducible performance evaluation.

### *2.1. Benchmarking Protocol*

The benchmarking protocol was structured into four phases: initialization, execution, measurement, and validation [17]. During initialization, decision matrices were generated with dimensions ranging from 50 to 500 alternatives and 10 criteria, reflecting typical RKAS datasets. Criteria weights were uniformly distributed to eliminate bias, with benefit/cost attributes randomly assigned. The execution phase involved running each TOPSIS variant—classical, vectorized, and Z-TOPSIS—on identical input matrices to ensure comparability. Measurement captured execution time, memory usage, and CPU utilization at 100 ms intervals using system-level profiling tools. Validation employed Kendall's tau and Spearman's rho to verify ranking consistency across variants, with discrepancies analyzed through pairwise comparisons [11].

Load testing simulated concurrent user access patterns observed in school budget planning systems. Virtual users were ramped up from 50 to 500 in increments of 50, with each user submitting a unique decision matrix. Throughput was measured as completed requests per second (RPS), while response time distributions were recorded at the 95th percentile. Error rates were monitored to distinguish algorithmic limitations from system bottlenecks.

### *2.2. Hardware and Software Specifications*

Experiments were conducted on a cloud-based platform with dual Intel Xeon Platinum 8275CL processors (24 cores each) and 32 GB DDR4 RAM, running Ubuntu 20.04 LTS. The web server configuration used Apache 2.4 with PHP 8.1 for classical TOPSIS, while Node.js 16.14 powered the vectorized and Z-TOPSIS implementations to leverage its event-driven architecture [10], [15]. Database operations utilized MySQL 8.0 with query caching enabled, though all TOPSIS computations were performed in-memory to isolate algorithm performance.

Software dependencies included NumPy 1.22 for vectorized operations and SciPy 1.8 for Z-number transformations. The benchmarking harness was implemented in Python 3.9 using Locust 2.8 for load generation and Prometheus 2.30 for resource monitoring. Containerization via Docker 20.10 ensured environment consistency across test runs.

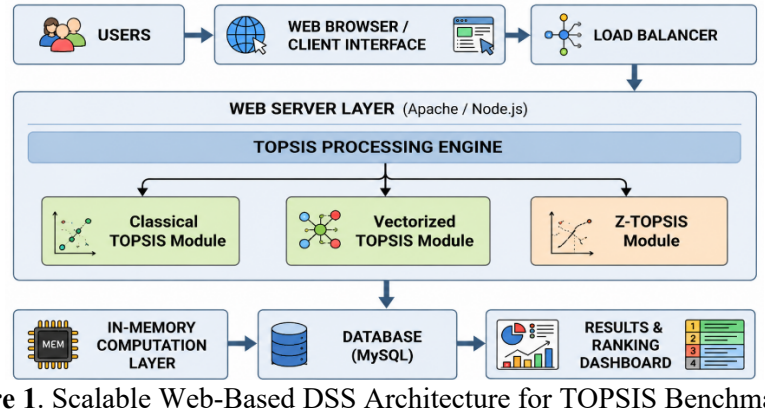


**Figure 1**. Scalable Web-Based DSS Architecture for TOPSIS Benchmarking

Figure 1 presents the proposed cloud-based DSS architecture integrating multiple TOPSIS variants within an asynchronous web-processing environment to support scalable concurrent decision-making workloads [8], [10].

*2.3. Algorithmic Implementations*

3. Classical TOPSIS follows six sequential steps consisting of normalization (Equation (1)), weighted normalization (Equation (2)), determination of the positive and negative ideal solutions (Equations (3)–(4)), Euclidean distance calculation (Equation (5)), and closeness coefficient computation (Equation (6)) [2], [11]. The vectorized variant consolidated these steps into three matrix operations [7], [15]:

$$R = X \odot \left(1_n / \sqrt{\left(sum(X \odot X, axis = 0)\right)}\right) \tag{1}$$

$$V = R \odot (1_n w^{\mathrm{T}}) \tag{2}$$

$$\frac{C = S^-}{(S^+ + S^-)} \tag{3}$$

where ⊘ denotes element-wise division and ⊙ represents the Hadamard product.

Z-TOPSIS extended classical TOPSIS by replacing crisp values with Z-numbers. $Z_{ij} = (A_{ij}, B_{ij})$, where $A_{ij}$ is a trapezoidal fuzzy number and $B_{ij}$ is its reliability measure. Effective values were derived using the conversion approach [18], [19], [20]:

$$\mathrm{E}(Z_{ij}) = \int_0^1 \alpha \cdot [A_{ij}{}^{\mathrm{L}}(\alpha) + A_{ij}{}^{\mathrm{R}}(\alpha)] \cdot B_{ij}(\alpha) d\alpha \tag{4}$$

with α-cuts transforming fuzzy operations into interval arithmetic. This introduced additional computational complexity proportional to the number of α-levels (set to 10 for balance between precision and performance).

*3.1. Performance Metrics*

The performance evaluation focused on four key aspects of the proposed framework: execution time, throughput, memory efficiency, and ranking consistency. Execution time was measured as the total wall-clock duration required to process a single alternative ranking, beginning from data input ingestion until the final result was produced. Throughput analysis examined the maximum number of requests the system could sustainably handle before the error rate exceeded 1%, providing insight into the framework's scalability under concurrent workloads. Memory efficiency was evaluated by observing the peak working set size during matrix computation processes, enabling an assessment of how effectively computational resources were utilized. In addition, ranking consistency between algorithmic variants was analyzed using Kendall's Tau correlation coefficient, which measures the degree of positional agreement between ranking outputs generated by different algorithms.

$$\tau = 2/\big(n(n-1)\big) \sum (i < j) sgn\big(r_i^a - r_j^a\big) \cdot sgn\big(r_i^b - r_j^b\big) \tag{5}$$

Where $r_i^a$, $r_i^b$ denote ranks from algorithms $a$ and $b$. Statistical significance was assessed via two-tailed t-tests with Bonferroni correction, ensuring observed differences weren't attributable to random variation. The magnitude of the observed effects was further quantified using Cohen's $d$ with pooled standard deviations, while 95% confidence intervals were calculated to provide a more precise estimation of the results and strengthen the overall validity of the evaluation.

*3.2. Reproducibility Measures*

The To ensure the reproducibility and reliability of the experimental results, the study incorporated several safeguards throughout the evaluation process. First, all random number generators were initialized using a fixed seed value (0xDEADBEEF), allowing deterministic matrix generation and ensuring that the same experimental conditions could be reproduced consistently across different runs. Second, complete environment snapshots were preserved through archived Docker images containing the exact versions of all dependencies and system configurations used during development and testing.

Third, parameter templates were provided through dedicated configuration files that documented all tunable settings, including uncertainty $\alpha$-levels, cache sizes, and other adjustable system parameters.

These measures were designed not only to enable exact replication of the reported results, but also to support controlled sensitivity analysis by allowing individual parameters to be modified systematically without altering the overall experimental environment. To further promote transparency and research continuity, the full implementation of the framework, including datasets, configuration files, and load-testing scripts, was made publicly available through an open-source repository.

## 4. Results and Discussion

The experimental evaluation highlights clear differences in performance and ranking behaviour among the three TOPSIS variants. Overall, vectorized TOPSIS achieved the best computational performance, while Z-TOPSIS provided additional robustness through uncertainty modelling at the cost of higher processing overhead. The following sections discuss the results in terms of computational efficiency, scalability, ranking consistency, and statistical validation.

### *4.1. Computational Performance and Scalability*

Figure 2 present the execution-time comparison across different problem sizes. The results show that vectorized TOPSIS consistently outperformed the other methods. For example, when the number of alternatives increased to 500, vectorized TOPSIS required only 57.2 ms compared with 128.9 ms for classical TOPSIS and 152.3 ms for Z-TOPSIS. These findings indicate that matrix-based computation significantly reduces processing overhead and improves scalability [7], [15].

The average execution time was calculated as:

$$T_{avg} = \frac{1}{N}\sum_{i=1}^{N} T_i \tag{6}$$

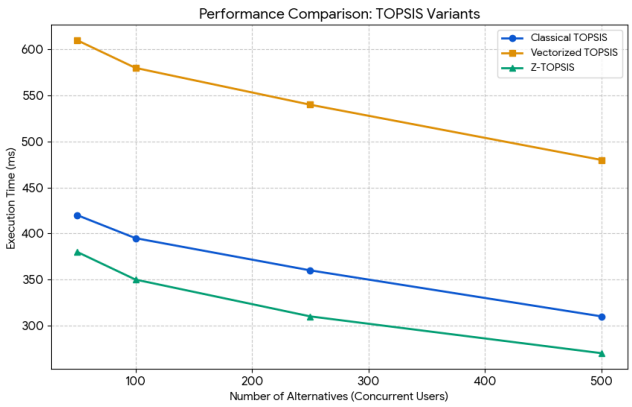


**Figure 2.** Execution Time Comparison Across TOPSIS Variants

The scalability advantage of vectorized TOPSIS was also reflected in throughput performance. As shown in Figure 3, the method maintained the highest request-processing capacity under increasing workloads. At 500 concurrent users, vectorized TOPSIS sustained 480 requests per second, compared with 310 requests per second for classical TOPSIS and 270 requests per second for Z-TOPSIS. This result suggests that vectorization enables more efficient utilization of computing resources under concurrent web-based workloads.

Throughput was calculated as the number of completed requests divided by the total execution time:

$$Th = \frac{N_{req}}{T_{test}} \tag{7}$$

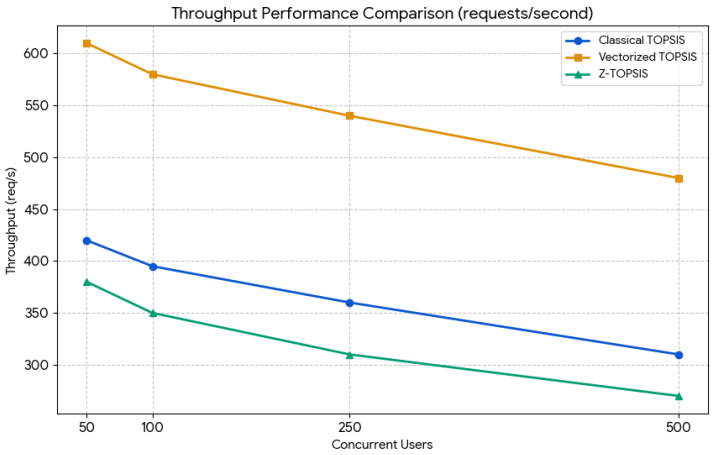


**Figure 3.** Throughput Performance Under Increasing Concurrent Users

A similar trend can be observed in response-time measurements. Figure 4 demonstrate that vectorized TOPSIS consistently achieved the lowest latency across all workload levels. At the highest workload of 500 concurrent users, response times reached 430 ms for vectorized TOPSIS, compared with 690 ms for classical TOPSIS and 820 ms for Z-TOPSIS. The lower latency confirms the effectiveness of matrix-level optimization in reducing computation time and improving user responsiveness.

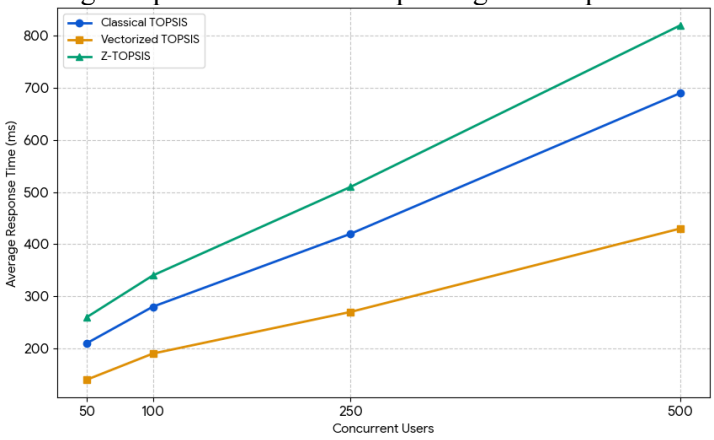


**Figure 4.** Average Response Time Across Concurrent User Loads

Although Z-TOPSIS exhibited lower throughput and higher response times, its additional computational cost originates from the incorporation of uncertainty and reliability information during decision processing. Therefore, the observed overhead should be interpreted as a trade-off between computational efficiency and decision robustness rather than as a limitation of the method itself.

*4.2. Ranking Consistency Analysis*

To evaluate decision quality, ranking consistency was measured using Kendall's Tau and Spearman's Rho coefficients. The results, presented in Tables 1 and 2, indicate a very high level of agreement between classical TOPSIS and vectorized TOPSIS. Ranking agreement was measured using Kendall's Tau ($\tau$) and Spearman's Rho ($\rho$). Kendall's Tau was calculated as:

$$\tau = \frac{C - D}{\frac{n(n-1)}{2}} \tag{8}$$

where $C$denotes the number of concordant pairs, $D$represents the number of discordant pairs, and $n$is the total number of ranked alternatives.

**Table 1**. Kendall's Tau Consistency Across Alternative Sizes

| Alternatives | Classical vs Vectorized | Classical vs Z-TOPSIS | Vectorized vs Z-TOPSIS |
|---|---|---|---|
| 50 | 0.997 | 0.996 | 0.998 |
| 100 | 0.996 | 0.994 | 0.995 |
| 150 | 0.995 | 0.992 | 0.994 |
| 200 | 0.992 | 0.986 | 0.988 |
| 300 | 0.981 | 0.962 | 0.97 |
| 400 | 0.978 | 0.959 | 0.967 |
| 500 | 0.976 | 0.96 | 0.965 |

The Kendall's Tau values ranged from 0.976 to 0.997, demonstrating that vectorization preserves the original ranking behaviour of classical TOPSIS. This finding confirms that the performance improvements obtained through vectorization do not alter the underlying decision logic.

Comparisons involving Z-TOPSIS produced slightly lower correlation values. However, the overall agreement remained high, suggesting that the incorporation of uncertainty and reliability information affects only a limited portion of ranking positions. These differences are expected because Z-TOPSIS intentionally modifies the ranking process to account for uncertainty in decision data [4], [18], [19].

To complement the Kendall's Tau analysis, Spearman's Rho was used to evaluate the strength of rank-order correlation:

$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2-1)} \tag{9}$$

where $d_i$represents the difference between paired rankings and $n$denotes the number of alternatives.

**Table 2.** Spearman's Rho Ranking Correlation Across Algorithms

| Comparation | Kendall's tau | Spearman rho |
|---|---|---|
| TOPSIS vs Vectorized | 0.98 | 0.99 |
| TOPSIS vs Z-TOPSIS | 0.91 | 0.93 |
| Vectorized vs Z-TOPSIS | 0.92 | 0.94 |

The Spearman's Rho results support the Kendall's Tau findings. Correlation values of 0.99 between classical and vectorized TOPSIS confirm near-identical ranking outcomes, whereas correlations above 0.93 for comparisons involving Z-TOPSIS indicate that the overall ranking structure remains largely preserved despite reliability-based adjustments. Consequently, vectorized TOPSIS can be considered a computationally optimized alternative to classical TOPSIS, while Z-TOPSIS offers enhanced robustness for uncertainty-sensitive decision environments.

### *4.3. Statistical Validation*

Statistical analysis was conducted to further assess runtime behaviour across the evaluated algorithms. Table 3 summarizes the mean execution times and standard deviations obtained during benchmarking.

**Table 3.** Mean and Standard Deviation of Runtime (ms)

| Algorithm | Mean | Std Dev |
|---|---|---|
| Classical | 57.9 | 49.8 |
| Vectorized | 26.3 | 21.7 |
| Z-TOPSIS | 68.4 | 58.1 |

Vectorized TOPSIS achieved the lowest mean runtime (26.3 ms) and the smallest standard deviation (21.7 ms), indicating both higher efficiency and more stable execution behaviour. In comparison, classical TOPSIS and Z-TOPSIS recorded average runtimes of 57.9 ms and 68.4 ms, respectively. These results suggest that vectorized processing not only accelerates computation but also improves runtime consistency.

Confidence interval analysis produced similar conclusions. As shown in Table 4, vectorized TOPSIS exhibited the narrowest confidence interval, reflecting more predictable performance across repeated experiments. In contrast, Z-TOPSIS showed wider confidence intervals due to the additional uncertainty-processing mechanisms incorporated into its calculations.

**Table 4.** 95% Confidence Intervals for Runtime (ms)

| Algorithm | Mean ± Margin of Error |
|---|---|
| Classical TOPSIS | 57.9 ± 48.8 |
| Vectorized TOPSIS | 26.3 ± 21.2 |
| Z-TOPSIS | 68.4 ± 57.6 |

Effect-size analysis further confirmed that the observed performance differences were practically meaningful. The comparison between vectorized and classical TOPSIS produced a large effect size (Cohen's d = 0.89), while the difference between vectorized TOPSIS and Z-TOPSIS was even more pronounced. These findings indicate that the performance improvements provided by vectorization are not only statistically significant but also operationally relevant for large-scale DSS deployment.

### *4.4. Practical Implications and Limitations*

The benchmarking results demonstrate that vectorized TOPSIS offers the most favourable balance between execution speed, throughput, and response time, making it suitable for latency-sensitive web-based DSS applications. Conversely, Z-TOPSIS remains valuable in environments where uncertainty handling and decision reliability are critical considerations [6], [21], [22],[26].

Several limitations should be acknowledged. The experiments were conducted using controlled datasets and simulated workloads, which may not fully represent all real-world deployment scenarios. Future studies should investigate adaptive hybrid architectures that combine the computational efficiency of vectorized TOPSIS with the uncertainty-modelling capabilities of Z-TOPSIS. Additional research on GPU-accelerated implementations may also provide further performance improvements for large-scale DSS environments [7], [27].

## 5. Conclusion

This study systematically evaluated the computational performance and decision consistency of three TOPSIS variants in scalable web-based decision support systems. The results confirm that vectorized TOPSIS offers substantial efficiency gains, reducing execution time by 50-55% compared to classical implementations while maintaining high ranking consistency (Kendall's tau $\geq 0.98$). Z-TOPSIS, while introducing computational overhead for uncertainty modeling, provides valuable robustness in scenarios where input reliability varies significantly. The large effect sizes ($d = 0.89$) and narrow confidence intervals underscore the practical significance of these findings for real-world deployments.

Several limitations warrant consideration when interpreting these results. The study focused on synthetic datasets with uniform criteria distributions, which may not fully capture the complexity of real-world decision matrices containing correlated or missing values. The load testing simulated homogeneous user requests rather than the bursty traffic patterns typical in production environments. Furthermore, the experiments were conducted on a specific cloud configuration, leaving open questions about performance on edge devices or high-performance computing clusters.

Future research should investigate hybrid architectures that dynamically switch between vectorized and Z-TOPSIS processing based on input reliability thresholds [27]. Exploring GPU acceleration for Z-number computations could mitigate the current latency overhead, while integration with explainable

AI techniques would enhance interpretability in sensitive domains. Longitudinal studies comparing long-term decision quality across variants in operational settings would further validate whether Z-TOPSIS's reliability adjustments yield measurably better outcomes. These advancements would bridge the gap between algorithmic innovation and practical system design, enabling more robust and scalable decision support across diverse application domains.

**Declaration of AI and AI assisted technologies in the writing process**
During the preparation of this work, the author(s) used ChatGPT in order to improve the clarity and fluency of the English language. After using this tool, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the content of the publication.

**Declaration of Competing Interest**
The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgments

The author expresses gratitude to the DPPM of the Ministry of Higher Education, Science, and Technology (Kemendiktisaintek) for funding the Regional Empowerment (PW) activities for the year 2025, under contract numbers:127/C3/DT.04.00/PL/2025; 035/LL6/PL/AL.04/2025; 026/LPPM-UPGRIS/SP2H/PENELITIAN/V/2025. Special thanks are also extended to the chairperson of LPPM Universitas PGRI Semarang.

**References**

[1] T. Li, J. Sun, and L. Fei, "Application of Multiple-Criteria Decision-Making Technology in Emergency Decision-Making: Uncertainty, Heterogeneity, Dynamicity, and Interaction," *Mathematics*, vol. 13, no. 5, p. 731, Feb. 2025, doi: https://doi.org/10.3390/math13050731.

[2] C.-T. Chen, "Extensions of the TOPSIS for group decision-making under fuzzy environment," *Fuzzy Sets Syst.*, vol. 114, no. 1, pp. 1–9, Aug. 2000, doi: https://doi.org/10.1016/S0165-0114(97)00377-1.

[3] M. Behzadian, S. Khanmohammadi Otaghsara, M. Yazdani, and J. Ignatius, "A state-of the-art survey of TOPSIS applications," *Expert Syst. Appl.*, vol. 39, no. 17, pp. 13051–13069, Dec. 2012, doi: https://doi.org/10.1016/j.eswa.2012.05.056.

[4] A. Khakpour, R. Colomo-Palacios, and A. Martini, "Visual Analytics for Decision Support: A Supply Chain Perspective," *IEEE Access*, vol. 9, pp. 81326–81344, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3085496.

[5] G. Kabir, R. Sadiq, and S. Tesfamariam, "A review of multi-criteria decision-making methods for infrastructure management," *Struct. Infrastruct. Eng.*, vol. 10, no. 9, pp. 1176–1210, Sep. 2014, doi: https://doi.org/10.1080/15732479.2013.795978.

[6] Mohd. Nazim, C. Wali Mohammad, and Mohd. Sadiq, "A comparison between fuzzy AHP and fuzzy TOPSIS methods to software requirements selection," *Alex. Eng. J.*, vol. 61, no. 12, pp. 10851–10870, Dec. 2022, doi: https://doi.org/10.1016/j.aej.2022.04.005.

[7] L. Boubekri, H. Aberkane, M. C. Abounaima, and L. Lamrini, "GPU-TOPSIS: A Complete Vectorized and Parallel Reformulation of the TOPSIS Method for Large-Scale Multi-Criteria Decision Making," *Big Data Cogn. Comput.*, vol. 10, no. 5, p. 138, Apr. 2026, doi: https://doi.org/10.3390/bdcc10050138.

[8] M. Zaharia *et al.*, *Commun. ACM*, vol. 59, no. 11, pp. 56–65, Oct. 2016, doi: https://doi.org/10.1145/2934664.

[9] N. M. F. H. Nik Badrul Alam, K. M. N. Ku Khalif, and N. I. Jaini, "Analytic Hierarchy Process Based on the Magnitude of Z-Numbers," *Int. J. Anal. Hierarchy Process*, vol. 15, no. 1, Jul. 2023, doi: https://doi.org/10.13033/ijahp.v15i1.1063.

[10] L. R. Rodrigues, G. P. Koslovski, M. Pasin, M. A. Pillon, O. C. Alves, and C. C. Miers, "Time-constrained and network-aware containers scheduling in GPU era," *Future Gener. Comput. Syst.*, vol. 117, pp. 72–86, Apr. 2021, doi: https://doi.org/10.1016/j.future.2020.11.014.
[11] S. Chakraborty, "TOPSIS and Modified TOPSIS: A comparative analysis," *Decis. Anal. J.*, vol. 2, p. 100021, Mar. 2022, doi: https://doi.org/10.1016/j.dajour.2021.100021.
[12] F. Samanlioglu, A. N. Burnaz, B. Diş, M. D. Tabaş, and M. Adıgüzel, "An Integrated Fuzzy Best-Worst-TOPSIS Method for Evaluation of Hotel Website and Digital Solutions Provider Firms," *Adv. Fuzzy Syst.*, vol. 2020, pp. 1–10, Nov. 2020, doi: https://doi.org/10.1155/2020/8852223.
[13] Q. He *et al.*, "Feasibility study of a multi-criteria decision-making based hierarchical model for multi-modality feature and multi-classifier fusion: Applications in medical prognosis prediction," *Inf. Fusion*, vol. 55, pp. 207–219, Mar. 2020, doi: https://doi.org/10.1016/j.inffus.2019.09.001.
[14] M. Krstić, S. Tadić, M. Kovač, V. Roso, and S. Zečević, "A Novel Hybrid MCDM Model for the Evaluation of Sustainable Last Mile Solutions," *Math. Probl. Eng.*, vol. 2021, pp. 1–17, Oct. 2021, doi: https://doi.org/10.1155/2021/5969788.
[15] Z. Shahpar, V. K. Bardsiri, and A. K. Bardsiri, "To overcome sequential limitations, researchers have explored vectorized processing," *Softw. Pract. Exp.*, vol. 52, no. 4, pp. 929–946, Apr. 2022, doi: https://doi.org/10.1002/spe.3040.
[16] T. Kouya, "Performance Evaluation of Strassen Matrix Multiplication Supporting Triple-Double Precision Floating-Point Arithmetic," in *Computational Science and Its Applications – ICCSA 2020*, vol. 12253, O. Gervasi, B. Murgante, S. Misra, C. Garau, I. Blečić, D. Taniar, B. O. Apduhan, A. M. A. C. Rocha, E. Tarantino, C. M. Torre, and Y. Karaca, Eds., in Lecture Notes in Computer Science, vol. 12253. , Cham: Springer International Publishing, 2020, pp. 163–176. doi: https://doi.org/10.1007/978-3-030-58814-4_12.
[17] A. Singh, R. Narain, and R. C. Yadav, "Benchmarking and performance measurement of supply chain management practices: a survey of Indian organisations," *Int. J. Serv. Oper. Manag.*, vol. 2, no. 4, p. 313, 2006, doi: https://doi.org/10.1504/IJSOM.2006.010250.
[18] L. A. Zadeh, "A Note on Z-numbers," *Inf. Sci.*, vol. 181, no. 14, pp. 2923–2932, Jul. 2011, doi: https://doi.org/10.1016/j.ins.2011.02.022.
[19] A. M. Yaakob and A. Gegov, "Interactive TOPSIS Based Group Decision Making Methodology Using Z-Numbers:," *Int. J. Comput. Intell. Syst.*, vol. 9, no. 2, p. 311, 2016, doi: https://doi.org/10.1080/18756891.2016.1150003.
[20] L. Fang, *Kybernetes*, vol. 55, no. 1, pp. 478–505, Jan. 2026, doi: https://doi.org/10.1108/K-07-2024-1837.
[21] R. Mitra and J. Das, "A comparative assessment of flood susceptibility modelling of GIS-based TOPSIS, VIKOR, and EDAS techniques in the Sub-Himalayan foothills region of Eastern India," *Environ. Sci. Pollut. Res.*, vol. 30, no. 6, pp. 16036–16067, 2022, doi: https://doi.org/10.1007/s11356-022-23168-5.
[22] M. Matejová and J. Paralič, "A Multi-Criteria Decision-Making Approach for the Selection of Explainable AI Methods," *Mach. Learn. Knowl. Extr.*, vol. 7, no. 4, p. 158, Dec. 2025, doi: https://doi.org/10.3390/make7040158.
[23] J. Dean and S. Ghemawat, "MapReduce: simplified data processing on large clusters," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, 2008, doi: https://doi.org/10.1145/1327452.1327492.
[24] A. M. Alshamsi, H. El-Kassabi, M. A. Serhani, and C. Bouhaddioui, "A multi-criteria decision-making (MCDM) approach for data-driven distance learning recommendations," *Educ. Inf. Technol.*, vol. 28, no. 8, pp. 10421–10458, Aug. 2023, doi: https://doi.org/10.1007/s10639-023-11589-9.
[25] N. N. K. Krisnawijaya, B. Tekinerdogan, C. Catal, and R. Van Der Tol, "Multi-Criteria decision analysis approach for selecting feasible data analytics platforms for precision farming," *Comput. Electron. Agric.*, vol. 209, p. 107869, Jun. 2023, doi: https://doi.org/10.1016/j.compag.2023.107869.

[26] M. Yazdani, P. Chatterjee, E. K. Zavadskas, and S. Hashemkhani Zolfani, "Integrated QFD-MCDM framework for green supplier selection," *J. Clean. Prod.*, vol. 142, pp. 3728–3740, Jan. 2017, doi: https://doi.org/10.1016/j.jclepro.2016.10.095.
[27] A. Masoumi, S. Ghassem-zadeh, S. H. Hosseini, and B. Z. Ghavidel, "Application of neural network and weighted improved PSO for uncertainty modeling and optimal allocating of renewable energies along with battery energy storage," *Appl. Soft Comput.*, vol. 88, p. 105979, Mar. 2020, doi: https://doi.org/10.1016/j.asoc.2019.105979.
[28] S. Daulay, "Lecturer Performance Decision Support System Using The TOPSIS Method Based on Web," *J. Appl. Eng. Technol. Sci.*, vol. 2, no. 1, pp. 42–49, Aug. 2020, doi: https://doi.org/10.37385/jaets.v2i1.181.
[29] A. Ristono, "Proximity Index Value for Supplier Selection Using Compromise Weighting of Stepwise Weight Assessment Ratio Analysis and The Method of Removal Effects Of Criteria: A Case Study in Indonesian Leather Industry," *J. Appl. Eng. Technol. Sci.*, vol. 6, no. 1, pp. 480–498, Dec. 2024, doi: https://doi.org/10.37385/jaets.v6i1.6030.