Two Weighting Methods for Software-Defined Network Controllers Ranking as a Multicriteria Decision-Making Problem

Firas Zobary[1*] and Li ChunLin[**]

School of Computer Science and Artificial Intelligence, Wuhan University of Technology
*Wuhan, Hubei, China (e-mail: *firas_zobary@hotmail.com, **chunlin74@aliyun.com*
[1]Corresponding Author: Firas Zobary

Abstract: Software Defined Networking (SDN) is a new paradigm designed to make networks manageable for network administrators. The controlling functions are managed by a controller which is the brain of SDN design and has a wide view of the network topology. SDN controllers have different features, so choosing a suitable controller for the network is a crucial challenge for administrators. Although there are many controller proposals for different kinds of networks in the literature, there is a limited comprehensive quantitative analysis for them. In this work, 14 of widely-used decentralized SDN controllers are evaluated and compared in terms of 8 criteria that each controller should possess. The comparison is done using 3 Multi-Criteria Decision-Making (MCDM) methods: Measurement of Alternatives and Ranking according to COmpromise Solution (MARCOS), Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS), and Multi-Attributive Ideal Real Comparative Analysis (MAIRCA) for a comprehensive comparative study. Moreover, two weighting methods: Entropy and MEthod based on the Removal Effects of Criteria (MEREC) are applied to assign an importance degree for each criterion. This study presents a novel approach to evaluate SDN controllers using a combination of 3 MCDM methods and 2 weight determination methods, resulting in six different ranking solutions. The most notable finding is that all six solutions identify the same two SDN controllers, Open Network Operating System (ONOS) and Open Day Light (ODL), as the best performers. These results demonstrate the effectiveness of using MCDM methods for comparing and evaluating SDN controllers.

Keywords: SDN, Control Plane, Decision Making, TOPSIS, MARCOS, MAIRCA

1. INTRODUCTION

The high growth of today's technologies has led researchers to rethink the current network architectures. Traditional networks start to show their limitations when new concepts like clouds and Internet of Things (IoT) are applied because of the reconfiguration difficulties. Almost every user and network vendor are looking forward to use SDN in different aspects. SDN is a new paradigm where the control plane is decoupled from the data plane. The controller is the main part of SDN design as it knows everything about the network and has a wide-view of the network topology. It communicates with the network devices and applications via Application Programming Interfaces (APIs) called "Southbound / Northbound Interfaces", and this communication is done using a standard protocol called OpenFlow (Alsaeedi et al., 2019).

NOX was the first SDN controller and later became a basic platform for many controllers such as POX, Ryu, and ONOS, and also many controllers' specifications have been developed over the past years. A centralized control plane has only one controller that is responsible for everything in the network, but at the same time, it is a single point of failure, whereas decentralized architecture is a better option as it contains multiple controllers to manage the network. In this paper, we focus on decentralized controllers due to their wide use and scalability features, then we make a comparative study and ranking according to predefined criteria.

The vast array of controllers, each with different properties, coupled with the growing demand for SDN implementation, makes the controller selection process a crucial challenge for administrators. These differences can significantly impact the decision-making process, prompting an important question "What is the optimal controller for deployment in the network based on service requirements?" This paper introduces a novel approach, not previously employed in the comparison and selection of SDN controllers, aimed at assisting network administrators in addressing this fundamental question.

As selecting a controller based on only one criterion is insufficient, we need a multicriteria selection, making it an MCDM problem. MCDM methods are strong mathematical tools for the comparative study of multialternative problems with different criteria.

This paper proposes 3 MCDM methods: TOPSIS, MARCOS, and MAIRCA for SDN controller comparative study and ranking. The combination of TOPSIS, MARCOS, and MAIRCA ensures a comprehensive and nuanced assessment, allowing for a well-rounded understanding of how each controller performs in comparison to the others. Furthermore, employing multiple methods provides a robustness check, enhancing the reliability and validity of the evaluation results. Moreover, we compare 14 well-known decentralized controllers based on 8 of the most important criteria each controller should possess before being selected.

We choose these 3 methods for 2 reasons: first, they prove their robustness and effectiveness compared to other MCDM methods (Badi and Pamucar, 2020), and the second reason is to validate our results, as using only one MCDM method may not be sufficient for controller selection. The main part of an MCDM is to assign weights to each criterion as a degree of importance. We apply 2 different weighting methods called Entropy and MEREC because they can distinguish different criteria weights more efficiently than other methods (Keshavarz-Ghorabaee, 2021). To the best of our knowledge, there is no work concerning a comprehensive comparative study among a wide number of SDN controllers and criteria using 3 MCDM and 2 weighting methods.

The main research contributions are as follows:

1) We compare 14 distributed controllers with 8 of the most important criteria that an SDN controller must possess before being adopted by a network designer.

2) To validate our results, we adopted 3 different MCDM methods: TOPSIS, MARCOS, and MAIRCA, and compared the controllers ranking and evaluation using different methods.

3) Criteria weighting assignment has been done using 2 different methods: Entropy and MEREC. The reason is to check the comparison results when the importance degree of criteria is changed.

The novelty of this study is that it is the first time for SDN controllers to be compared using 3 MCDM and 2 weighting methods. Moreover, using more than one MCDM method to compare SDN controllers may provide a more comprehensive and robust analysis than using only one method, and also contribute to the growing body of research on SDN and network management.

The study's emphasis on practical usability is evident in the convergence of results. The use of MCDM methods provides a systematic and objective framework for decision-making in the selection of SDN controllers. This aspect is crucial for practical applications, as it helps stakeholders make informed decisions based on a well-defined set of criteria rather than relying solely on subjective opinions or isolated performance metrics. The study's applicability extends beyond a mere ranking of controllers; it offers actionable insights for network architects, operators, and organizations involved in SDN deployment. In conclusion, the insights gained from this research not only contribute to the academic understanding of SDN controllers' performance but also offer tangible benefits for practitioners seeking reliable, effective, and practical solutions for their network infrastructure.

The organization of the paper is as follows: Section 2 reviews the related works and previous studies. In Section 3, the MCDM methods are introduced. Section 4 conducts the SDN controller comparison study, while Section 5 applies MCDM to the data. Results and discussion are introduced in Section 6, followed by the conclusion in Section 7.

2. RELATED WORKS

MCDM methods are attracted by researchers because of their wide use in many engineering and industrial directions. TOPSIS is one of the most used methods (Çelikbilek and Tüysüz, 2020) and was developed by Hwang and Yoon (Hwang et al., 1993). Authors in (Durkadevi et al., 2022) used TOPSIS and Analytic Hierarchy Process (AHP) in SDN controller selection. Entropy-based TOPSIS is proposed in (Kannan and Thiyagarajan, 2022) to select a controller among 4 alternatives: POX, Floodlight, Beacon, and Ryu. Authors in (Ider and Barekatain, 2021) also used TOPSIS and AHP to select a controller concerning 3 criteria: Central Processing Unit (CPU) utilization, incoming flows rate, and the number of hops between switch and controller. However, these criteria are not enough to make the best selection. An MCDM called Best-Worst Method (BWM) is done in (Amiri et al., 2020) but most of their alternatives are centralized single controllers which make the selection not suitable for SDN scalability using multicontroller architecture. Authors in (Ali et al., 2022) investigated several SDN controllers and compared them with an analytical network process approach based only on performance in Mininet emulator. AHP method has been used in (AlShehri and Mishra, 2019) where the authors compared 5 SDN controllers: POX, Ryu, Trema, Floodlight, and ODL, by mapping the criteria values to values in a predefined scale, and "Ryu" is selected as the best controller concerning these criteria. The study conducted by (Vani and RamaMohanBabu, 2023) introduces a dynamic load-balancing algorithm for SDN based on server response time and content mapping. The proposed method is subjected to a comparative analysis against other load balancing techniques, including round-robin and random methods. The comparison results demonstrate a significant improvement of 58% when employing MCDM with equal criteria weights. Additionally, there is a 50% enhancement observed when utilizing MCDM with unequal weights assigned to various Quality of Service (QoS) parameters, compared to other load balancing methods. An MCDM method called Analytical Network Decision-making Process (ANDP) was employed in (Ali and Roh, 2022) to choose the optimal SDN controller for SDN-IoT. The findings revealed that ONOS controller has the best features for SDN-IoT.

It is noteworthy that some works exclusively compare SDN controllers based only on performance metrics such as latency and throughput, neglecting their diverse properties, which leads to a lack of accuracy in the comparison process. The study by (Shahzad et al., 2023) selected the optimal controller based on performance and cost calculated mathematically and Non-dominated Sorting Genetic Algorithm II (NSGA-II) was employed to select the controller to reduce migrations and control communications. In the work by (Hadi et al., 2023), the authors conducted an assessment of 14 SDN controllers by examining their architectures and identifying their strengths and weaknesses. However, the assessment lacked an in-depth analysis or benchmarking of the controllers. In (Nxumalo et al., 2017) the authors evaluated the performance of Onix and Kandoo based on emulation. They concluded that Kandoo has less complexity whereas Onix is better in large networks, but this study didn't mention other controllers. In (Singh et al., 2022) ODL and ONOS were compared and ONOS outperformed ODL based on throughput metric. A performance evaluation was done in (Zhu et al., 2019) using 3 benchmarking tools to

compare 9 controllers and concluded that ODL and ONOS have better flow installation rates than others, but benchmarking should be supported with other criteria before selecting a controller to be adopted. (Çil and Demirci, 2024) compared 4 SDN controllers (Ryu, ONOS, ODL, and POX) in terms of network forensics issues such as packet size, source-target detection, attack and attacker information, and the impact of Denial of Service (DoS) attacks. Notably, the data collection was solely from the southbound interface concerning forensics in SDN. Authors in (Sharma and Verma, 2023) conducted a performance comparison of Ryu and floodlight controllers using Mininet emulator. Meanwhile, (Rashid et al., 2023) conducted a performance evaluation of SDN controllers in wired and wireless networks. The authors evaluated two controllers, POX and Ryu, using Mininet-WiFi emulator and the Distributed Internet Traffic Generator (D-ITG) to assess the aforementioned controllers based on delay, jitter, packet loss, and throughput metrics. The results indicated that Ryu consistently outperformed POX, particularly in terms of lower latency, jitter, packet loss, and higher throughput, especially in wireless networks.

The most of previous literatures related to SDN adopted TOPSIS and AHP with only limited number of controllers and criteria and they used only one weighting method to identify the criteria importance degree, which is not enough to make a comprehensive comparison and make a decision about which controller to be the most suitable for our network. Also, MARCOS and MAIRCA are not used for SDN controller selection before. For this reason, this study gives a high accuracy of the obtained results as it combines 3 MCDM methods and 2 weighting methods, as well as 14 alternatives according to 8 criteria.

3. MCDM METHODS

3.1 Criteria weights calculation

First, it is essential to differentiate between two types of criteria: benefit and cost criteria. Benefit criteria have a positive impact with higher values indicating better performance. Conversely, cost criteria have a negative impact, where lower values are indicative of superior performance. In our study, all the criteria are categorized as benefit criteria, and none are designated as cost criteria.

1) Method 1: Entropy

Entropy from (Shannon, 2001) can be used to assign weights to criteria. Using entropy, the importance degree of each criterion is related to information generated by alternatives evaluation under this criterion. Where m is the number of alternatives, n is the number of criteria, $i=1,..,m$, and $j=1,..,n$, the main steps of entropy method are as follows:

Step 1: Normalize the decision matrix D to get normalized matrix R:

$$r_{ij} = \frac{d_{ij}}{\sum_{k=1}^{m} d_{kj}} \tag{1}$$

Step 2: Calculate the entropy value for each criterion:

$$e_j = \frac{-\sum_{i=1}^{m} \ln(r_{ij})}{\ln(m)} \tag{2}$$

Step 3: Calculate the variation degree of each criterion:

$$v_j = |1 - e_j| \tag{3}$$

Step 4: Extract the criteria weights:

$$w_j = \frac{v_j}{\sum_{k=1}^{n} v_k} \tag{4}$$

2) Method 2: MEREC

This method is a new approach for weights calculation. Its main idea is to remove the effect of criteria in the decision matrix to determine the importance degree of those criteria (Keshavarz-Ghorabaee et al., 2021).

Step 1: Normalize the decision matrix D to get normalized matrix R

$$r_{ij} = \begin{cases} \frac{\min_k (d_{kj})}{d_{ij}} & if \ j \in C^+ \\ \frac{d_{ij}}{\max_k (d_{kj})} & if \ j \in C^- \end{cases} \tag{5}$$

where C^+ is the benefit criteria and C^- is the cost criteria set and $k = 1,..,m$.

Step 2: Calculate the performance of each alternative:

$$S_i = \ln\left[1 + \left(\frac{1}{n}\sum_j |\ln(r_{ij})|\right)\right] \tag{6}$$

Step 3: Calculate the performance of each alternative concerning the removal of each criterion.

$$S'_{ij} = \ln\left[1 + \left(\frac{1}{n}\sum_{k,k\neq j} |\ln(r_{ik})|\right)\right] \tag{7}$$

Step 4: Calculate the removal effect of each criterion.

$$E_j = \sum_i |S'_{ij} - S_i| \tag{8}$$

Step 5: Determine the criteria weights.

$$w_j = \frac{E_j}{\sum_{j=1}^{n} E_j} \tag{9}$$

3.2 Comparative methods

1) MARCOS

This method is well-suited for our study as it enables a detailed analysis of how each SDN controller is compared to the ideal solution in terms of the selected criteria. By incorporating the order of preferences, MARCOS provides a robust framework for evaluating and ranking the controllers based on their overall performance. The following are the steps to apply MARCOS:

Step1: Initiate the decision matrix that includes the alternatives and their criteria. Each intersection between an alternative and a criterion is given as d_{ij}.

$$D=\begin{pmatrix} d_{11} & \cdots & d_{1n} \\ \vdots & \ddots & \vdots \\ d_{m1} & \cdots & d_{mn} \end{pmatrix}$$

Step 2: Determine the Ideal and Anti-Ideal solutions

$$AI=\left\{\left\langle \max(d_{ij})\right\rangle, j\in C^{+} \mid \left\langle \min(d_{ij})\right\rangle, j\in C^{-}\right\} \tag{10}$$

$$AAI=\left\{\left\langle \min(d_{ij})\right\rangle, j\in C^{+} \mid \left\langle \max(d_{ij})\right\rangle, j\in C^{-}\right\} \tag{11}$$

this will generate an extended decision matrix:

$$D^{''}=\begin{array}{c} \\ AAI \\ A_1 \\ \vdots \\ A_m \\ AI \end{array}\begin{bmatrix} d_{aa1} & d_{aa2} & \cdots & d_{aan} \\ d_{11} & d_{12} & \cdots & d_{1n} \\ \vdots & \vdots & \cdots & \vdots \\ d_{m1} & d_{m2} & \cdots & d_{mn} \\ d_{a1} & d_{a2} & \cdots & d_{an} \end{bmatrix}$$

Step 3: Normalize the extended decision matrix to get a normalized matrix:

$$r_{ij}=\begin{cases} \frac{d_{ij}}{d_{ai}} \text{ if } j\in C^{+} \\ \frac{d_{ai}}{d_{ij}} \text{ if } j\in C^{-} \end{cases} \tag{12}$$

Step 4: Multiply the weight with the normalized matrix to form the weighted normalized matrix $V=\left[v_{ij}\right]$

$$v_{ij}=r_{ij}\times w_j \tag{13}$$

Step 5: Calculate the utility degrees of each alternative K_i^{+} and K_i^{-}, the relationships between the ideal and anti-ideal solutions, respectively.

$$K_i^{+}=\frac{\sum_{j=1}^{n} v_{ij}}{\sum_{j=1}^{n} v_{aj}} \tag{14}$$

$$K_i^{-}=\frac{\sum_{j=1}^{n} v_{ij}}{\sum_{j=1}^{n} v_{aaj}} \tag{15}$$

Step 6: Determine the utility functions $f\left(K_i^{+}\right)$ and $f\left(K_i^{-}\right)$ related to the ideal and anti-ideal solutions.

$$f\left(K_i^{+}\right)=\frac{K_i^{-}}{K_i^{+}+K_i^{-}} \tag{16}$$

$$f\left(K_i^{-}\right)=\frac{K_i^{+}}{K_i^{+}+K_i^{-}} \tag{17}$$

Step 7: Calculate the utility function of each alternative.

$$f\left(K_i\right)=\frac{K_i^{+}+K_i^{-}}{1+\frac{1-f\left(K_i^{+}\right)}{f\left(K_i^{+}\right)}+\frac{1-f\left(K_i^{-}\right)}{f\left(K_i^{-}\right)}} \tag{18}$$

Step 8: Rank the alternatives based on the utility function. The desired alternative is the one with the highest $f\left(K_i\right)$.

2) TOPSIS

TOPSIS is chosen for its simplicity and effectiveness in determining the best alternative from a set of options. TOPSIS is particularly suitable for our evaluation of SDN controllers, as it offers a straightforward way to compare controllers based on multiple criteria, providing decision-makers with a comprehensible and easily interpretable ranking. The following are the steps to apply TOPSIS:

Step 1: Create the decision matrix as step 1 in MARCOS.

Step 2: Normalize the decision matrix to create the normalized decision matrix.

$$r_{ij}=\frac{d_{ij}}{\sqrt{\sum_{i=1}^{m} d_{ij}^2}} \tag{19}$$

Step 3: Calculate the weighted normalized matrix $V=\left[v_{ij}\right]$.

$$v_{ij}=r_{ij}\times w_j \tag{20}$$

Step 4: Determine the positive and negative ideal solutions A^{+} and A^{-}.

$$A^{+}=\left\{\left\langle \max(v_{ij})\right\rangle, j\in C^{+} \mid \left\langle \min(v_{ij})\right\rangle, j\in C^{-}\right\} \tag{21}$$

$$A^{-}=\left\{\left\langle \min(v_{ij})\right\rangle, j\in C^{+} \mid \left\langle \max(v_{ij})\right\rangle, j\in C^{-}\right\} \tag{22}$$

Step 5: Calculate the Euclidean distance between each alternative and bot the positive and negative ideal solutions

$$E_i^{+}=\sqrt{\sum_{j=1}^{n}\left(v_{ij}-A_j^{+}\right)^2} \tag{23}$$

$$E_i^{-}=\sqrt{\sum_{j=1}^{n}\left(v_{ij}-A_j^{-}\right)^2} \tag{24}$$

Step 6: Calculate the degree of similarity for each alternative.

$$\mu_i=\frac{E_i^{-}}{E_i^{+}+E_i^{-}} \tag{25}$$

Step 7: Rank the alternatives according to the similarity degree as the selected alternative is the one with the greatest degree.

3) MAIRCA

MAIRCA is selected due to its ability to handle multi-attribute decision problems and its consideration of both real and ideal alternatives. This method considers the deviation between real alternatives and the ideal solution, providing a nuanced understanding of the controllers' performance. Given

the complexity of SDN controller evaluation with multiple criteria, MAIRCA allows a comprehensive analysis that considers the trade-offs and compromises inherent in real-world decision-making. The following are the steps to apply MAIRCA:

Step 1: Create the decision matrix as step 1 in MARCOS and TOPSIS.

Step 2: Create the alternative preference matrix $P_{m\times 1}$ in which the elements are the preference of each alternative. We assumed that there is no preferred alternative as the decision maker has no priority for an alternative over the others, so: $p_{11} = p_{12} = \cdots = p_{m1} = \frac{1}{m}$, and the preference matrix will be formed as:

$$P = \begin{bmatrix} p_{11} \\ p_{12} \\ p_{13} \\ \vdots \\ p_{m1} \end{bmatrix}$$

Step 3: Determine the theoretical ponder matrix T_t by multiplying the alternative preference matrix by the criteria weight matrix.

$$t_{tij} = p_{i1} \times w_j \tag{26}$$

Step 4: Calculate the actual ponder matrix $T_a = [t_{amn}]$ where:

$$t_{aij} = t_{tij} \times \frac{d_{ij} - d_j^-}{d_j^+ - d_j^-}\ ,\ j \in C^+ \tag{27}$$

$$t_{aij} = t_{tij} \times \frac{d_{ij} - d_j^+}{d_j^- - d_j^+}\ ,\ j \in C^- \tag{28}$$

and:

$$D^+ = \left\{ \left\langle \max(d_{ij}) \right\rangle, j \in C^+ \mid \left\langle \min(d_{ij}) \right\rangle, j \in C^- \right\} \tag{29}$$

$$D^- = \left\{ \left\langle \min(d_{ij}) \right\rangle, j \in C^+ \mid \left\langle \max(d_{ij}) \right\rangle, j \in C^- \right\} \tag{30}$$

so:

$$D^+ = \left\{ d_1^+, d_2^+, \cdots, d_n^+ \right\} \tag{31}$$

$$D^- = \left\{ d_1^-, d_2^-, \cdots, d_n^- \right\} \tag{32}$$

Step 5: Create the gap matrix in which the elements can be extracted as the difference between the theoretical ponder matrix and the actual ponder matrix.

$$g_{ij} = t_{tij} - t_{aij} \tag{33}$$

Step 6: Calculate the final criterion function in which the elements are the sum of the gaps for each alternative.

$$Q_i = \sum_{j=1}^{n} g_{ij} \tag{34}$$

Step 7: Rank the alternatives according to their final criterion function, as the desirable alternative is the one with the lowest Q.

4. SDN CONTROLLERS' COMPARISON STUDY

4.1 Controllers' investigation methodology

The methodology aims to comprehensively view the most well-known and widely-used controllers and capture their properties. In this study, we focus only on decentralized controllers because centralized controller is a single point of failure and doesn't bring scalability advantage to the network. The investigation covers 14 decentralized controllers. First of all, we check the journals, conferences, and white papers in which these controllers are described and record each property we found in a table. Second, we explore the official website of each controller and record its property values in the table. Third, we collect more properties and their values from the technical talks, workshops, and conferences and involve them in our comparison table. To ensure accuracy, certain properties are cross verified using different resources.

4.2 Qualitative criteria

Different criteria should be assessed to evaluate the controller to select an SDN controller. We choose 8 of the most important criteria among which we compare our controllers. Those criteria are as follows:

- Programming language: different programming languages are used to write SDN controllers, such as Python, Java, C, C++, Go, Haskell, and Erlang. Some controllers have been written using one programming language, whereas others have been written using multiple languages. Every programming language has its own properties, and the controller will inherit these properties as well. In addition, controllers with multiple languages can behave more efficiently regarding memory allocation and execution on multiple platforms, thus they got higher evaluation than controllers that use only one programming language. Moreover, different programming languages have different evaluation. By assessing controllers based on programming languages, the evaluation captures inherent properties associated with different programming languages. This criterion ensures a thorough understanding of how programming language choices impact the overall behavior and functionality of SDN controller.

- SouthBound Interface (SBI): these interfaces are responsible for communication between the controller and SDN-enabled network devices. SDN controllers use different versions, and some controllers can support multiple versions. More OpenFlow versions brings higher evaluation to the controller in terms of this criteria. Evaluating controllers based on their SBI compatibility ensures that the chosen controllers can effectively communicate with a variety of SDN-enabled devices, promoting interoperability and adaptability in dynamic network environments.

- Platform: this property describes the controller compatibility with different platforms and operating systems. Most SDN controllers are built on top of Linux, and some support Windows and MacOS. Assessing controllers based

on their compatibility with different platforms ensures that they can seamlessly integrate into diverse network environments. This criterion reflects the need for controllers to be adaptable and functional across various operating systems. In this study, the more platform the controller has, the better evaluation it has in terms of platform criteria.

- License and Documentation: most SDN controllers are considered open source, but some have a proprietary license and can only be pre-requested only for research purposes. This proprietary for some controllers is a challenge for developers so that they cannot get regular updates as open-source controllers. This criterion emphasizes the importance of choosing controllers that align with open standards, promoting transparency, accessibility, and continued support. During our investigation, we found that some controllers lack documentation whereas others are updated regularly so we gave them higher evaluation in terms of documentation criteria as they are easier to be used by administrators and designers.

- Multithreading: this property is very important for scalability problems. Single-threading controllers are suitable for small-size networks, whereas multithreading controllers are more suitable for big-size networks such as 5G and SD-WAN networks. Controller multithreading refers to the ability of an SDN controller to process multiple requests simultaneously using multiple threads of execution. Multithreading can improve the performance and scalability of an SDN controller by allowing it to process requests in parallel and take advantage of multiple CPU cores. SDN controllers must also be designed to handle concurrent access to shared data structures and avoid race conditions and deadlocks. This criterion ensures that the selected controllers are equipped to meet the performance and scalability requirements of diverse network environments.

- Modularity: The modularity of the controller enables the convergence of various programs. In a distributed system, a controller with high modularity will execute tasks faster. SDN controller modularity provides a flexible and scalable framework for managing network resources. It enables organizations to easily add new functionality, upgrade or replace components, and adapt to changing business requirements. This criterion reflects the need for controllers that can seamlessly integrate into distributed network architectures, supporting dynamic and evolving network requirements.

- Consistency: Controller consistency is an important aspect of SDN controller design. It refers to the ability of the controller to maintain a consistent view of the network state and ensures that all network devices are configured correctly according to the desired network policies. The controller's consistency reflects its ability to adapt to network changes and faults. Consistency is essential for controllers to provide a coherent and reliable view of the network, aligning with the desired policies and configurations. Some controllers have good and solid consistency against network failures, whereas others have poor adaption to network changes.

4.3 Comparative study settings

In this subsection, 14 controllers are selected as our comparison study's alternatives. In Table 1, we comprehensively view our alternatives according to the abovementioned criteria.

For SDN controllers' comparison, qualitative criteria were assigned values to assess their levels (Amiri et al., 2020). We set these values as: (1=limited, 2=fair, 3=good, 4= high), so a decision matrix is generated to be used for MCDM methods in which each row presents an alternative and each column represents a criterion as they appear in Table 2.

5. COMPARISON STUDY USING MCDM METHODS

In this section, we propose 3 different MCDM methods (TOPSIS, MARCOS, and MAIRCA) to rank the 14 SDN controllers according to 8 important criteria every controller should possess before being applied in the network. We also combine these MCDM methods with 2 different weighting methods (Entropy and MEREC) to calculate the weight for each criterion, reflecting its importance in the comparative process. The following subsections introduce the weights calculation and the results of the three MCDM methods as well as the ranking of alternatives according to each method.

5.1 Criteria weights calculation

Assigning weights for our criteria is done according to the steps in section 3 using two different methods. Using the entropy method, a decision matrix is normalized using (1), then an entropy value and variation degree for each criterion is calculated by (2) and (3). Finally, (4) is used to calculate criteria weights. In the second weighting method (MEREC), the decision matrix is normalized as in (5). Next, we find the performance of each alternative by using (6) and (7), respectively. The last step is to calculate the removal effect of each criterion and criteria weights using (8) and (9). The criteria weights found using these two methods are illustrated in 5.3 TOPSIS

Using TOPSIS method, the decision matrix is normalized by (19), and after the weighted normalized matrix is calculated, the positive and negative ideal solutions A^+ and A^- are determined by (21) and (22). Then, (23) and (24) are used to calculate the distance between each alternative and both A^+ and A^-. The final result for each alternative's degree of similarity is calculated using (25).

5.4 MAIRCA

This method determines a theoretical and actual ponder matrix using (26) and (27). Then, we create the gap matrix and calculate the final criterion functions for each alternative using (33) and (34). Table **6** shows the results for each alternative.

Table 5 shows the values of TOPSIS method for each alternative.

Table 3 which shows the differences between criteria weights as different weighting methods use different data to extract the weights.

5.2 MARCOS

First, the positive and negative ideal alternatives are found using (10) and (11) respectively. After the decision matrix normalizing and getting the weighted normalized matrix, we calculate the alternative utility degrees K_i^+ and K_i^-. Next, the values of $f\left(K_i^+\right)$ and $f\left(K_i^-\right)$ are determined according to (16) and (17). It was found that with the entropy weighting method, $f\left(K_i^+\right)=0.744$ and $f\left(K_i^-\right)=0.256$, and with MEREC weighting method, $f\left(K_i^+\right)=0.75$ and $f\left(K_i^-\right)=0.25$. Finally, we get the utility function of each alternative using (18). Table 4 shows the final utility function values for each alternative with respect to both Entropy and MEREC weighting methods

Table 1. SDN Controllers Comprehensive View.

	Prog. Language	**SBI**	**Platform**	**License**	**Multithreading**	**Modularity**	**Consistency**	**Documentation**
Beehive (Yeganeh & Ganjali, 2016)	Go	1.0,1.2	Linux	Apache2.0	Yes	Good	Yes	Limited
Disco (Phemius et al., 2014)	Erlang	1.0	Linux, MacOS, Widows	Proprietary	No	Good	No	Limited
HyperFlow (Liao et al., 2019)	C++	1.0	Linux, MacOS, Widows	Proprietary	Yes	Fair	No	Limited
Kandoo (Hassas Yeganeh & Ganjali, 2012)	C, C++, Python	1.0-1.02	Linux	Proprietary	Yes	High	No	Limited
Loom (Zhu et al., 2020)	Erlang	1.3-1.4	Linux	Apache2.0	Yes	Good	No	Good
Onix (Mattos et al., 2016)	Python	1.0, OVSDB	Linux	Proprietary	Yes	Good	No	Limited
ONOS (Berde et al., 2014)	Java	1.0,1.3	Linux, MacOS, Widows	Apache2.0	Yes	High	Yes	Good
ODL (Belkadi et al., 2019)	Java	1.0,1.3	Linux, MacOS, Widows	EPL 1.0	Yes	High	Yes	Good
OpenIRIS (Lee et al., 2014)	Java	1.0-1.3	Linux	Apache2.0	Yes	Fair	No	Limited
PANE (Ferguson et al., 2013)	Haskell	1.0	Linux, MacOS	BSD 3.0	No	Fair	No	Fair
RunOS (Shalimov et al., 2015)	C++	1.3	Linux	Apache2.0	Yes	High	Yes	Fair
SmartLight (Botelho et al., 2014)	Java	1.3	Linux	Proprietary	No	-	No	Limited
Yanc (Monaco et al., 2013)	C, C++	1.0-1.3	Linux	Proprietary	No	-	No	Limited
ZeroSDN (Kohler et al., 2018)	C++	1.0,1.3	Linux	Apache2.0	No	High	Yes	Fair

Table 2. Decision Matrix Containing Qualitative Data.

	Prog. Language	**SBI**	**Platform**	**License**	**Multithreading**	**Modularity**	**Consistency**	**Documentation**
Beehive	1	3	2	3	3	3	3	1
Disco	1	2	4	1	1	3	1	1
HyperFlow	2	2	4	1	3	2	1	1
Kandoo	4	3	2	1	3	4	1	1
Loom	1	4	2	3	3	3	1	3

Onix	2	4	2	1	3	3	1	1
ONOS	3	3	4	3	3	4	3	3
ODL	3	3	4	3	3	4	3	3
OpenIRIS	3	3	2	3	3	2	1	1
PANE	1	2	4	3	1	2	1	2
RunOS	2	2	2	3	3	4	3	2
SmartLight	3	2	2	1	1	1	1	1
Yanc	3	3	2	1	1	1	1	1
ZeroSDN	2	3	2	3	1	4	3	2

5.3 TOPSIS

Using TOPSIS method, the decision matrix is normalized by (19), and after the weighted normalized matrix is calculated, the positive and negative ideal solutions A^+ and A^- are determined by (21) and (22). Then, (23) and (24) are used to calculate the distance between each alternative and both A^+ and A^-. The final result for each alternative's degree of similarity is calculated using (25).

5.4 MAIRCA

This method determines a theoretical and actual ponder matrix using (26) and (27). Then, we create the gap matrix and calculate the final criterion functions for each alternative using (33) and (34). Table **6** shows the results for each alternative.

Table **5** shows the values of TOPSIS method for each alternative.

Table 3. Criteria Weights Using Entropy and MEREC.

	Entropy	MEREC
C1	0.123	0.159
C2	0.039	0.14
C3	0.08	0.053
C4	0.158	0.131
C5	0.135	0.153
C6	0.104	0.21
C7	0.199	0.077
C8	0.154	0.077

Table 4. Results and Ranking Using MARCOS.

	Entropy		MEREC	
	$f(K_i)$	Rank	$f(K_i)$	Rank
Beehive	0.658	4	0.658	6
Disco	0.387	11	0.465	11
HyperFlow	0.475	10	0.475	10
Kandoo	0.552	7	0.688	4
Loom	0.638	5	0.69	3
Onix	0.479	9	0.587	8
ONOS	0.869	1	0.849	1
ODL	0.869	1	0.849	1
OpenIRIS	0.571	6	0.635	7
PANE	0.504	8	0.484	9
RunOS	0.747	2	0.733	2
SmartLight	0.361	13	0.382	13
Yanc	0.37	12	0.414	12
ZeroSDN	0.674	3	0.672	5

5.4 MAIRCA

This method determines a theoretical and actual ponder matrix using (26) and (27). Then, we create the gap matrix and calculate the final criterion functions for each alternative using (33) and (34). Table **6** shows the results for each alternative.

Table 5. Results and Ranking Using TOPSIS.

	Entropy		MEREC	
	μ_i	Rank	μ_i	Rank
Beehive	0.534	4	0.5	7
Disco	0.201	13	0.317	10
HyperFlow	0.3	10	0.374	12
Kandoo	0.422	6	0.646	2
Loom	0.488	5	0.523	6
Onix	0.307	9	0.496	8
ONOS	0.865	1	0.796	1
ODL	0.865	1	0.796	1
OpenIRIS	0.42	7	0.53	5
PANE	0.363	8	0.319	9
RunOS	0.67	2	0.626	3
SmartLight	0.234	12	0.294	13
Yanc	0.241	11	0.319	11
ZeroSDN	0.608	3	0.577	4

Table 6. Results and Ranking Using MAIRCA.

	Entropy		MEREC	
	Q_i	Rank	Q_i	Rank
Beehive	0.03	4	0.031	6
Disco	0.061	11	0.058	11
HyperFlow	0.05	9	0.048	9
Kandoo	0.044	7	0.029	4
Loom	0.032	5	0.026	3
Onix	0.051	10	0.037	8
ONOS	0.004	1	0.009	1
ODL	0.004	1	0.009	1
OpenIRIS	0.04	6	0.034	7
PANE	0.046	8	0.051	10
RunOS	0.02	2	0.024	2
SmartLight	0.065	13	0.064	13
Yanc	0.064	12	0.059	12
ZeroSDN	0.028	3	0.03	5

6. RESULTS AND DISCUSSION

The comparative results using 3 MCDM methods, each with 2 criteria weight assignment proposals are introduced in Table 7 and Table 8, and from these results, we observe the following:

1. The 14 controllers are ranked differently when different MCDM methods are applied.

2. Despite variations in rankings, ONOS and ODL controllers are ranked as the best SDN controllers in all the three MCDM methods applied based on 8 of the most important criteria the controller must possess, regardless of the proposed criteria weights assignment method, which means that selecting the best alternative is independent of the applied MCDM method, ensuring the accuracy of the selection.

3. Comparing the results of the methods with both weighting methods concludes that the criteria importance level (i.e., criteria weights) can affect the ranking of each MCDM method at a different rate. In MARCOS method, 7 of 14 alternatives have the same ranking even though different weighting methods are applied, with a similarity rate of 50%. At the same time, it is 28.5% (4 of 14) for TOPSIS and only 21.4% (3 of 14) for MAIRCA, making MARCOS has a high stability for SDN controller selection problem dominating TOPSIS which is used by most literatures.

4. Using entropy for assigning criteria weights, 7 of 14 alternatives have the same ranking with respect to MARCOS, TOPSIS and MAIRCA. However, using MEREC method, only 4 of 14 have the same ranking.

The practical implications of the above results can be highlighted through a series of case studies that illustrate the real-world applications of SDN controller evaluation. In an enterprise network upgrade scenario, our evaluation guides the selection of SDN controllers, specifically ONOS and ODL, ensuring a seamless transition to enhance scalability and overall network management. For a telecommunication service provider, this research aids in the selection of controllers that meet criteria crucial for supporting high-speed data services, ensuring efficient network resource management. Additionally, in the context of cloud service and smart city implementation, our findings guide the choice of SDN controllers to optimize resources and support diverse applications, facilitating efficient service delivery and infrastructure management. Furthermore, this study demonstrates its applicability in enhancing research and education networks, aiding data center network management for increased efficiency, scalability, and adaptability. These case studies collectively underscore the practical relevance and versatility of our evaluation results in diverse real-world scenarios.

While our study endeavors to comprehensively evaluate SDN controllers using a combination of MCDM methods and weighting techniques, it is essential to acknowledge certain limitations. The selected criteria for evaluation, though carefully chosen, might not encompass all aspects crucial for practical deployment. Factors such as energy efficiency, adaptability to emerging technologies, or specific industry requirements might not have been explicitly considered. Furthermore, potential biases could exist in the weighting methods employed, and the subjectivity inherent in determining the importance of each criterion should be recognized. The generalizability of our results may also be subject to certain limitations. The study primarily focuses on widely used decentralized SDN controllers, and the applicability of the findings to specific network architectures, sizes, or industries might vary. Researchers must interpret the results within the context of their unique network environments.

Table 7. SDN Controllers Ranking Using MERERC.

	MEREC		
	MARCOS	**TOPSIS**	**MAIRCA**
Beehive	6	6	7
Disco	11	11	10
HyperFlow	10	9	12
Kandoo	4	4	2
Loom	3	3	6
Onix	8	8	8
ONOS	1	1	1
ODL	1	1	1
OpenIRIS	7	7	5
PANE	9	10	9
RunOS	2	2	3
SmartLight	13	13	13
Yanc	12	12	11
ZeroSDN	5	5	4

Table 8. SDN Controllers Ranking Using Entropy.

	ENTROPY		
	MARCOS	**TOPSIS**	**MAIRCA**
Beehive	4	4	4
Disco	11	13	11
HyperFlow	10	10	9
Kandoo	7	6	7
Loom	5	5	5
Onix	9	9	10
ONOS	1	1	1
ODL	1	1	1
OpenIRIS	6	7	6
PANE	8	8	8
RunOS	2	2	2
SmartLight	13	12	13
Yanc	12	11	12
ZeroSDN	3	3	3

Finally, the future of SDN controllers is marked by transformative trends and promising research avenues. Integrating machine learning into controllers enhances adaptability and intelligence, while heightened emphasis on security aligns with evolving cyber threats. The rise of 5G networks calls for seamless integration, and edge computing prompts tailored solutions for efficient resource management. Standardization efforts and interoperability are crucial for cohesive networking ecosystems. Energy-efficient SDN controllers align with sustainability, and the exploration of Intent-Based Networking principles streamlines network management. Navigating these trends and research directions is pivotal for advancing SDN controllers, providing essential insights for researchers and practitioners alike.

7. CONCLUSION

In this work, 14 SDN controllers of decentralized control plane architecture has been studied in detail and compared to each other using different properties. The comparative study was done according to 8 of the most important criteria the SDN controller must possess before being chosen for the network. For comparison and ranking, 3 MCDM methods: MARCOS, TOPSIS, and MAIRCA were applied for the MCDM process. Moreover, 2 methods: Entropy and MEREC were applied for criteria weights determination which gives each criterion an importance degree. The conclusions from the results are as follows:

1. Compared to the previous literatures, this is the first time 3 different MCDM methods with 2 different criteria weighting methods are applied to compare and rank SDN decentralized controllers.

2. ODL and ONOS controllers are chosen as the best SDN controllers in all MCDM and weighting methods making MCDM a valid way for designers and administrators to select the best SDN controller.

3. Comparing between MARCOS, TOPSIS and MAIRCA with respect to the weighting methods, in MARCOS method, 7 of 14 alternatives have the same ranking even though different weighting methods are applied, with a similarity rate of 50%. At the same time, it is 28.5% (4 of 14) for TOPSIS and only 21.4% (3 of 14) for MAIRCA, making MARCOS the most stable MCDM method for SDN controller selection problem dominating TOPSIS which is used by most literatures.

4. The comparison between entropy and MEREC shows that entropy is more accurate than MEREC as 7 of 14 alternatives have the same ranking with respect to MARCOS, TOPSIS and MAIRCA, while only 4 of 14 have the same ranking using MEREC method.

5. A performance evaluation for each controller can be done in the future to strengthen the reliability of the study results.

REFERENCES

Analysis. In *SDN-Supported Edge-Cloud Interplay for Next Generation Internet of Things* (pp. 105–124). Chapman and Hall/CRC.

Alsaeedi, M., Mohamad, M. M., & Al-Roubaiey, A. A. (2019). Toward adaptive and scalable OpenFlow-SDN flow control: A survey. *IEEE Access*, *7*, 107346–107379.

AlShehri, M. A. R., & Mishra, S. (2019). Feature based comparison and selection of SDN controller. *International Journal of Innovation and Technology Management*, *16*(05), 1950029.

Amiri, E., Alizadeh, E., & Rezvani, M. H. (2020). Controller selection in software defined networks using best-worst multi-criteria decision-making. *Bulletin of Electrical Engineering and Informatics*, *9*(4), 1506–1517.

Badi, I., & Pamucar, D. (2020). Supplier selection for steelmaking company by using combined Grey-MARCOS methods. *Decision Making: Applications in Management and Engineering*, *3*(2), 37–48.

Belkadi, O., Laaziz, Y., Vulpe, A., & Halunga, S. (2019). An Integration of OpenDaylight and OpenNebula for Cloud Management Improvement using SDN. *2019 27th Telecommunications Forum (TELFOR)*, 1–4.

Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O'Connor, B., Radoslavov, P., & Snow, W. (2014). ONOS: towards an open, distributed SDN OS. *Proceedings of the Third Workshop on Hot Topics in Software Defined Networking*, 1–6.

Botelho, F., Bessani, A., Ramos, F. M. V, & Ferreira, P. (2014). On the design of practical fault-tolerant SDN controllers. *2014 Third European Workshop on Software Defined Networks*, 73–78.

Çelikbilek, Y., & Tüysüz, F. (2020). An in-depth review of theory of the TOPSIS method: An experimental analysis. *Journal of Management Analytics*, *7*(2), 281–300.

Çil, A., & Demirci, M. (2024). A comparative analysis of software-defined network controllers in terms of network forensics processes and capabilities. *Sigma Journal of Engineering and Natural Sciences*, *42*(3).

Durkadevi, K., Revathi, T., & Shenbagalakshmi, G. (2022). Generic Method for SDN Controller Selection Using AHP and TOPSIS Methods. *International Journal of Information Technology & Decision Making*, *21*(03), 1031–1059.

Ferguson, A. D., Guha, A., Liang, C., Fonseca, R., & Krishnamurthi, S. (2013). Participatory networking: An API for application control of SDNs. *ACM SIGCOMM Computer Communication Review*, *43*(4), 327–338.

Hadi, F. E., Al-Kaseem, B. R., & Al-Raweshidy, H. S. (2023). Comprehensive Exploration and Performance Assessment of SDN Controller. *Al-Iraqia Journal for Scientific Engineering Research*, *2*(4), 83–90.

Hassas Yeganeh, S., & Ganjali, Y. (2012). Kandoo: a framework for efficient and scalable offloading of control applications. *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, 19–24.

Hwang, C.-L., Lai, Y.-J., & Liu, T.-Y. (1993). A new approach for multiple objective decision making. *Computers & Operations Research*, *20*(8), 889–899.

Ider, M., & Barekatain, B. (2021). An enhanced AHP–TOPSIS-based load balancing algorithm for switch migration in software-defined networks. *The Journal of Supercomputing*, *77*(1), 563–596.

Kannan, D., & Thiyagarajan, R. (2022). Entropy based TOPSIS method for controller selection in software defined networking. *Concurrency and Computation: Practice and Experience*, *34*(1), e6499.

Keshavarz-Ghorabaee, M. (2021). Assessment of distribution center locations using a multi-expert subjective–objective decision-making approach. *Scientific Reports*, *11*(1), 1–19.

Keshavarz-Ghorabaee, M., Amiri, M., Zavadskas, E. K., Turskis, Z., & Antucheviciene, J. (2021). Determination of objective weights using a new method based on the removal effects of criteria (MEREC). *Symmetry*, *13*(4), 525.

Kohler, T., Dürr, F., & Rothermel, K. (2018). ZeroSDN: A highly flexible and modular architecture for full-range distribution of event-based network control. *IEEE Transactions on Network and Service Management*, *15*(4), 1207–1221.

Lee, B., Park, S. H., Shin, J., & Yang, S. (2014). IRIS: the Openflow-based recursive SDN controller. *16th International Conference on Advanced Communication Technology*, 1227–1231.

Liao, L., Lai, C.-F., Wan, J., Leung, V. C. M., & Huang, T.-C. (2019). Scalable distributed control plane for On-line social networks support cognitive neural computing in software defined networks. *Future Generation Computer Systems*, *93*, 993–1001. Zhu,

Mattos, D. M. F., Duarte, O. C. M. B., & Pujolle, G. (2016). A resilient distributed controller for software defined networking. *2016 IEEE International Conference on Communications (ICC)*, 1–6.

Monaco, M., Michel, O., & Keller, E. (2013). Applying operating system principles to SDN controller design. *Proceedings of the Twelfth ACM Workshop on Hot Topics in Networks*, 1–7.

Nxumalo, M. N., Mba, I., & Adigun, M. O. (2017). Comparative study for control plane scalability approaches in SDN productive networks. *2017 International Conference on Engineering and Technology (ICET)*, 1–6.

Phemius, K., Bouet, M., & Leguay, J. (2014). DISCO: Distributed SDN controllers in a multi-domain environment. *2014 IEEE Network Operations and Management Symposium (NOMS)*, 1–2.

Rashid, S. J., Alkababji, A. M., & Khidhir, A. S. M. (2023). Performance evaluation of software-defined networking controllers in wired and wireless networks. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, *21*(1), 49–59.

Shahzad, M., Liu, L., Belkout, N., & Antonopoulos, N. (2023). Optimal controller selection and migration in large scale software defined networks for next generation internet of things. *SN Applied Sciences*, *5*(12), 309.

Shalimov, A., Nizovtsev, S., Morkovnik, D., & Smeliansky, R. (2015). The runos openflow controller. *2015 Fourth European Workshop on Software Defined Networks*, 103–104.

Shannon, C. E. (2001). A mathematical theory of communication. *ACM SIGMOBILE Mobile Computing and Communications Review*, *5*(1), 3–55.

Sharma, A., & Verma, G. S. (2023). Performance comparison of Ryu and floodlight SDN controller. *AIP Conference Proceedings*, *2705*(1).

Singh, A., Kaur, N., & Kaur, H. (2022). Extensive performance analysis of OpenDayLight (ODL) and Open Network Operating System (ONOS) SDN controllers. *Microprocessors and Microsystems*, *95*, 104715.

Vani, K. A., & RamaMohanBabu, K. N. (2023). An Intelligent Server load balancing based on Multi-criteria decision-making in SDN. *International Journal of Electrical and Computer Engineering Systems*, *14*(4), 433–442.

Yeganeh, S. H., & Ganjali, Y. (2016). Beehive: Simple distributed programming in software-defined networks. *Proceedings of the Symposium on SDN Research*, 1–12.

Zhu, L., Karim, M. M., Sharif, K., Li, F., Du, X., & Guizani, M. (2019). SDN controllers: Benchmarking & performance evaluation. *ArXiv Preprint ArXiv:1902.04491*.

L., Karim, M. M., Sharif, K., Xu, C., Li, F., Du, X., & Guizani, M. (2020). SDN controllers: A comprehensive analysis and performance evaluation study. *ACM Computing Surveys (CSUR)*, *53*(6), 1–40.