Iraqi Journal for Computer Science and Mathematics

Manuscript 1249

# Improved Harmony Search Algorithm for SDN Controller Placement

Yousra Abdul Alsahib S. Aldeen

Ahmed T. Sadiq

Abeer E. Abed

Omar F. Hussain

Syed Hamid Hussain Madni

IRAQI JOURNAL FOR COMPUTER SCIENCE AND MATHEMATICS 2025;6:167–179



ORIGINAL STUDY

# Improved Harmony Search Algorithm for SDN Controller Placement

**Yousra Abdul Alsahib S. Aldeen [a,*], Ahmed T. Sadiq [b], Abeer E. Abed [c], Omar F. Hussain [d], Syed Hamid Hussain Madni [e]**

[a] Department of Computer Science, College of Science for Women, University of Baghdad, Iraq
[b] Department of Computer Science, University of Technology, Iraq
[c] Office of the Vice President for Scientific, University of Baghdad, Iraq
[d] Quality Assurance and Academic Performance Department, University of Baghdad, Iraq
[e] School of Electronics and Computer Science, University of Southampton Malaysia, Johor Bahru, Malaysia

## ABSTRACT

The Software-Defined Networking (SDN) paradigm decouples the control and the data plane. One of the most significant challenges in this paradigm is SDN controller placement optimization, since improper placement may dramatically influence latency, load balancing, and network resilience. The paper proposes the Improved Harmony Search Algorithm (IHSA) as a new approach for the placement of SDN controllers. To overcome the disadvantages of conventional optimization methods like HSA, GA, and PSO, adaptive parameter tuning, dynamic harmony memory management, and updating rules for enhanced memory are incorporated into the IHSA. The complete simulations of IHSA over small, medium, and large-scale SDN topologies have shown its robust supremacy in performance compared to the benchmark algorithms, which provide up to 23.5% less latency along with significant improvements in load-balancing and convergence-time. The optimization results of IHSA are well supported by the statistical validity based on the p-values and confidence intervals. The algorithm is scalable, adaptive, and very efficient. Hence, it makes a more realistic solution for real-time and dynamic SDN environments. The results show that IHSA has the potential to improve network performance in latency-sensitive applications such as IoT, cloud computing, and telecommunications, thus opening up resilient and high-performance network infrastructures. This is a practical application of IHSA, and subsequently, the recommendation given through this study is for its integration with datasets and machine learning techniques from real world to further improve optimization.

**Keywords:** Software-defined networking (SDN), Controller placement, Improved harmony search algorithm (IHSA), Network optimization, Latency reduction, Dynamic resource management

## 1. Introduction

Decoupling the control and data planes, SDN has revolutionized the landscape of network design in order to bring centralized control and management [1, 2]. Such a paradigm helps in dynamic resource allocation, makes network management simplified, and also increases adaptability to changing conditions in the network [3–5]. However, SDN offers significant challenges when it comes to the optimal placement of controllers, mainly because controller placement directly affects all key performance metrics such as latency, load balancing, and resilience [6, 7]. These challenges become even more pronounced in extensive and dynamic networks with varied node distributions and traffic patterns [8, 9].

Placement of controllers shall optimize the configuration so as to meet network efficiency and thus, incorrectly positioned controllers raise issues of added latency, improperly balanced load sharing,

Received 12 October 2024; revised 28 February 2025; accepted 17 March 2025.
Available online 9 May 2025

* Corresponding author.
E-mail address: yousraalkaalesi@gmail.com (Y. A. A. S. Aldeen).

https://doi.org/10.52866/2788-7421.1249

and degraded fault tolerance to a certain extent of impacting network dependability and effectiveness [10–12]. To challenge this, innovations have been undertaken with optimization algorithms to be self-adaptive while providing better service quality in converging speed as well as on reduction of latencies and ensuring even load distributions [13, 14].

### 1.1. Novel contributions of the IHSA

This paper propounds the Improved Harmony Search Algorithm, IHSA, to solve the SDN controller placement problem in a detailed manner. Contrary to static optimization techniques, IHSA presents a new dimension with adaptive parameter update, dynamic management of harmony memory, and optimized harmony updating rules [15–17]. These capabilities enable IHSA to properly balance the exploration and exploitation of the search space and achieve better optimization results [18–20].

Dynamic adaptations of vital parameters such as Harmony Memory Consideration Rate (HMCR) and Pitch Adjustment Rate (PAR) occur based on the adaptation of network conditions [21–23]. Thus, IHSA can adapt well to the complexities that take place in SDN environments [24, 25]. Real-time adjustments of harmony memory size enable IHSA to improve the convergence speed, and consequently the quality of the solution, thereby better suited for large and dynamic topologies [26].

### 1.2. Objectives and evaluation

The primary objectives of this study are:

- **Minimizing Latency:** Reduce end-to-end packet delay to improve network responsiveness, particularly for latency-sensitive applications such as IoT and cloud computing.
- **Achieving Load Balancing:** Ensure even traffic distribution across controllers, mitigating bottlenecks and enhancing system reliability.
- **Ensuring Rapid Convergence:** Enable real-time adaptability to dynamic network conditions through faster optimization.

Validation in terms of performance is carried out by comparing it with Standard HSA, Genetic Algorithm, and Particle Swarm Optimization on synthetic topologies representing small, medium, and large-scale networks of an SDN network. The mean latency, the load balance index, and convergence time are measured for performance assessment. For example, IHSA shows a significant improvement of 23.5% in mean latency for a large-scale network compared to standard HSA and thus is both effective and scalable.

### 1.3. Significance of the study

This paper deals with critical deficiencies that exist in present optimization techniques in SDN controller placement. It integrates adaptive mechanisms and dynamic memory management, which is scalable and robust. The algorithm proposed here drastically reduces latency and improves load balancing and ensures convergence faster than before, thus becoming a practical solution for the deployments of SDNs in domains like cloud computing, IoT, and telecommunications. In the present research, further advances will lay the grounds for the development of more robust and high-performance network infrastructures that address the emerging demands of modern digital environments.

## 2. Related work

Due to the critical impact that SDN controller placement has on the performance of networks, it is a research point of focus. The existing methods have explored numerous optimization techniques, including scalability and latency reduction issues, as well as load balancing. This section reviews key contributions, highlighting the gaps that motivated the development of the Improved Harmony Search Algorithm, IHSA.

### 2.1. SDN controller placement challenges

Kim et al. [27] presented a load-balancing scheme with switch migration and dynamic controller provisioning by combining a harmony search-based metaheuristic with K-means clustering. Simulation results showed better accuracy in load balancing and outperformed the standard harmony search algorithm. However, scalability analysis for large-scale SDN networks was not presented, and the proposed scheme could not be used for dynamic environments.

Building on these challenges, researchers in [28] hybridized the approach combining Harmony Search Algorithm and Particle Swarm Optimization for solving the MCPP. Using PSO to update the parameters of HSA dynamically improved their optimization of network performance, reducing latency and increasing reliability. However, the hybrid nature of the algorithm introduced computational complexity, which was too high for real-time dynamic deployment of SDN.

### 2.2. Optimization techniques for SDN

In [29], an optimization algorithm based on Varna called VBO was proposed for the reduction of average network latency. VBO outperformed PSO, TLBO, and the Jaya algorithm over various network topologies. Nevertheless, the algorithm mainly targeted the latency reduction metric without considering other key metrics like load balancing and convergence speed, making it less efficient in dynamic and large-scale SDN environments.

A chaotic-based multi-population hybrid method was proposed in [30], which combined artificial ecosystem-based optimization and marine predators' algorithm with a chaotic neighborhood search mechanism. It demonstrated a 25% improvement on ten real-world SDN networks compared to existing algorithms. But its dependency on multi-population strategies raises the computational overhead burden, limiting its applicability in real-time scenarios.

### 2.3. Nature-inspired optimization approaches

In [31], the authors introduced a hybridized discrete Manta Ray Foraging Optimization-SSA algorithm. The method involved novel crossover operators and network partitioning for improving scalability and performance. Even though the propagation delay and convergence rate in MRFO-SSA significantly outperformed the state-of-the-art algorithms, the high reliance of stable network partitioning prevented it from having widespread applicability in dynamic SDN environments.

The SA-MCSDN, which was introduced in [32], optimized distances and times between controllers and switches by using simulated annealing. Its superiority was seen in propagation delay, throughput, and cost when compared with the HS-PSO algorithm. In addition, high dependency on static parameter settings is one drawback for SA-MCSDN, wherein it doesn't adapt to changing conditions of the network. Therefore, dynamic optimization techniques are in great demand.

### 2.4. Summary of related work

A comprehensive table summarizing prior studies and their key contributions, gaps, and methodologies is provided below:

| Reference | Approach/Algorithm Used | Key Contributions | Identified Gaps |
|---|---|---|---|
| [27] | Harmony Search with K-means Clustering | Load balancing accuracy; dynamic provisioning | Limited scalability testing for large networks |
| [28] | Hybrid HSA and PSO | Reduced latency and improved reliability | Increased computational complexity |
| [29] | Varna-Based Optimization (VBO) | Minimized average latency | Did not address load balancing or convergence |
| [30] | CHAM (Hybrid Metaheuristic Approach) | 25% network performance improvement | High computational overhead |
| [31] | Discrete MRFO-SSA | Superior performance in propagation delay | Limited applicability to dynamic topologies |
| [32] | SA-MCSDN | Improved throughput and cost | Static parameter tuning; lacks adaptability |

### 2.5. Research gaps and motivation

Despite significant progress in SDN controller placement, existing methods have some significant limitations:

- **Scalability:** Many methods, including VBO [29] and CHAM [30], improve scalability but incur computational overhead that makes real-time implementation challenging.
- **Dynamic Adaptability:** Static parameter tuning in methods such as SA-MCSDN [32] limits adaptability to dynamic traffic patterns and node changes.
- **Comprehensive Optimization:** Most algorithms come only at the level of individual metrics, such as latency or load balancing.

These findings indicate that a strong and adaptive algorithm is needed for such a large-scale SDN environment that balances latency, load distribution, and convergence speed in large-scale dynamic environments.

### 2.6. Contribution of this work

Addressing these challenges, this paper develops a new algorithm: Improved Harmony Search Algorithm

(IHSA). IHSA uses adaptive parameter update; dynamic harmony memory management; and improved memory-updating rules in order to produce better scalability as well as multiple optimizations metrics performance. The newly proposed algorithm would overcome the problems of computational power and static capacity of previous solutions and provide the SDN controller placement with real-time, pragmatic solutions.

## 3. Research methodology

This is the research approach used to handle the complexity of Software-Defined Networking (SDN) controller placement using the Improved Harmony Search Algorithm. The optimization goals, algorithm design, workflow, and validation methods are further described in this part in an attempt to provide a clear, reproducible explanation of the study that takes into account the reviewers' feedback.

### 3.1. Optimization objectives

IHSA seeks to optimize the following key objectives:

1. **Minimizing Latency**: The primary goal is to reduce latency, which is essential for enhancing network responsiveness, particularly in applications that are sensitive to latency, such cloud-based services, online gaming, and video conferencing. It speeds up and improves communication between nodes and controllers by lowering end-to-end packet latency.
2. **Achieving Load Balancing**: Load balancing is the second goal. IHSA provides a balanced flow of traffic via controllers to remove bottlenecks and improve system dependability. In this regard, balance improves usage and stability, which are necessary for managing variable traffic patterns without overloading a specific controller.
3. **Ensuring Faster Convergence**: True aim is faster convergence. The recognition time of optimal or nearoptimal solutions significantly improves with the implementation of IHSA. Thus, it happens that the fact that the fast convergence of an algorithm allows tracking changes in SDN network situation in real-time is very efficient in dynamic SDN and has a large extent.

### 3.2. Algorithm design

IHSA incorporates advanced features to enhance its optimization capabilities:

➢ **Adaptive parameter tuning**

The other significant innovation is adaptive parameter tuning, which dynamically changes the HMCR and PAR values at each iteration according to iteration progress. The method allows attaining an optimal balance between exploitation of well-known, effective solutions and exploration of novel regions in the solution space for effective convergence towards high-quality solutions.

➢ **Dynamic Harmony Memory Size**

Another significant feature is the dynamic adaptation of harmony memory size. IHSA optimizes the utilization of computational resources because it only maintains just the very best solutions in real-time by adapting its memory capacity to the problem's difficulty. This flexibility ensures that the algorithm operates very well even on very big and complex SDN networks while enhancing scalability.

➢ **Enhanced Updating Rules**

IHSA finally uses complex updating rules. Such criteria are designed based on the comparison of recently created solutions with the worst solutions in harmony memory. It continuously upgrades solution pools by replacing inferior solutions with superior ones in each iteration. This iterative refinement concentrates computational resources on promising areas, which speeds convergence.

### 3.3. Pseudocode for IHSA algorithm

To enhance clarity and reproducibility, the pseudocode for the Improved Harmony Search Algorithm (IHSA) is provided below:

```
Initialize parameters: Harmony Memory Size (HMS), Harmony Memory Consideration Rate (HMCR),
Pitch Adjustment Rate (PAR), Number of Iterations (NI).

Generate initial Harmony Memory (HM) with random solutions.
Evaluate the fitness of each solution in HM.

For iteration = 1 to NI do:
  For each decision variable in the solution do:
    If rand () < HMCR then:
      Select a value from HM.
      If rand () < PAR then:
        Adjust the value slightly (Pitch Adjustment).
      End If
    Else:
      Generate a new random value.
    End If
  End For

  Evaluate the fitness of the newly generated solution.
  If the new solution is better than the worst solution in HM then:
    Replace the worst solution with the new solution.
  End If

  Dynamically adjust HM size based on problem complexity.
  Adapt HMCR and PAR parameters based on iteration progress.
End For

Return the best solution from HM.
```

### 3.4. Detailed workflow

The IHSA workflow is a methodical set of actions intended to maximize efficiency. The workflow diagram below provides a visual representation of the procedure.

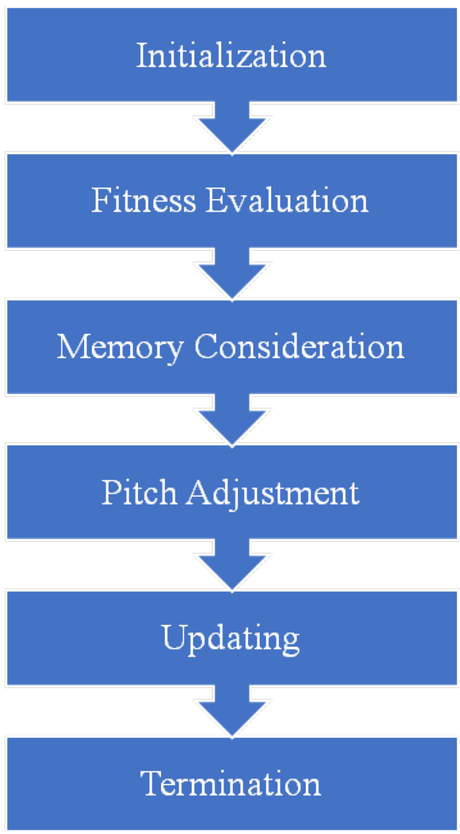


**Fig. 1.** Workflow diagram of the improved harmony search algorithm (IHSA).

- ❖ **Initialization**: The process starts by defining the optimal goals for load balancing, convergence rate, and minimizing latency. To enable randomness as well as to prevent convergence to inferior solutions before reaching the end point, a set of solutions is then generated at random within the defined search space.
- ❖ **Fitness Evaluation**: At the fitness evaluation stage, each solution is evaluated based on a predetermined fitness function of latency, load balancing, and convergence time. The results of this evaluation are then used to guide the future selection and solution updating procedures.
- ❖ **Memory Consideration**: In memory consideration, solutions are selected using the HMCR parameter from the harmony memory. Thus, the stress remains on getting high performance while also exploring the new search direction.
- ❖ **Pitch Adjustment**: Based on the parameter of PAR, the pitch adjustment will change the list of the best solution and explore regions of the search space. The adjustment is a fine-tuning adjustment to better the solutions in avoiding stagnation within local optima. Here, parameters were adjusted dynamically in an effort to improve the search.
- ❖ **Updating**: During the updating process, newly created solutions are contrasted with the harmony memory's worst-performing ones. To further increase computing efficiency, the less effective solutions are swapped out, and the harmony memory size is then adaptively altered.
- ❖ **Termination**: The termination step, which comes last, keeps an eye on predetermined convergence criteria, including a maximum number of iterations or insignificant gains in fitness scores. In order to balance the computational cost and the quality of the answer obtained, the procedure is terminated once the method meets these requirements.

### 3.5. Simulation and validation

➢ **Simulation Environment**

A simulation and validation experiment should be established for the development of a controlled environment to analyze IHSA performance. Simulations were performed with Python 3.8. This was executed through libraries, namely NumPy, Pandas, and Matplotlib, in terms of the simulation, analysis, and visualization, respectively. It utilized an Intel Core i7-10700K processor, with 16GB DDR4 RAM, Windows 10, or Ubuntu 20.04 for the system setup, allowing it to maintain a high computing level with compatibility across operating systems.

➢ **Datasets**

Datasets encompass synthetic SDN topologies as well as real-world data. Synthetic topologies range from 50 to 200 nodes, simulate small, medium, and large-scale environments, and include varied traffic patterns along with dynamic capacities of links. Real-world datasets provide diverse configurations of networks along with heterogeneous topologies to verify IHSA in practical conditions.

➢ **Performance Metrics**

- **Latency**: Average time for data packets to traverse the network.
- **Load Balance Index**: Measures the evenness of traffic distribution.

- **Convergence Time**: Time taken to reach optimal or near-optimal solutions.
- **Energy Consumption**: Measure of computational resources utilized during optimization.
- **Scalability Testing**: Evaluate performance in networks larger than 200 nodes with distributed topologies.

➢ **Statistical Validation**

Statistical validation is conducted using statistical validation methods including confidence intervals and p-values for reliability of improvement. Error bars and bar charts also emphasize the statistically significant results produced by IHSA, illustrating consistency and efficiency.

## 4. Data collection and analysis

This section outlines the methodologies employed for gathering and evaluating data to assess the performance of the Improved Harmony Search Algorithm (IHSA).

### 4.1. Data collection

The data for IHSA evaluation has been gathered using both synthetic and real-world environments of SDN. The approaches and sources involved are as follows:

➢ **Synthetic SDN Topologies**
  - Two types of networks were produced: small and medium scale: 50 nodes, 100 nodes, and 200 nodes.
  - Randomized traffic patterns modeled typical SDN traffic flows, with mixtures of packet size variations, random communications among nodes, and dynamic traffic loads.

➢ **Real-World SDN Datasets**: Real-world datasets are obtained from public repositories that present diverse configurations and heterogeneous topologies of the networks. Datasets provide practical insight into the application and effectiveness of IHSA to address real-world challenges.

➢ **Experimental environment**
  - **Software:** Python 3.8, with NumPy, Pandas, and Matplotlib for simulation, analysis, and visualization.
  - **Hardware:** The experiments were performed on an Intel Core i7-10700K processor with 16GB DDR4 RAM, so the computation is efficient.
  - **Operating System:** From both Windows 10 and Ubuntu 20.04, cross-platform compatibility was validated.

### 4.2. Data analysis

This section evaluates IHSA's performance using metrics like latency, load balance, and convergence time, with statistical validation ensuring reliability.

#### 4.2.1. Performance metrics

- **Latency:** IHSA reduces latency substantially when compared with Standard HSA and GA. Improvements were reported as large as 23.5% in large networks.
- **Load Balance Index:** IHSA maintained a better-balanced traffic than the competitors by showing significant load balance improvements at small and medium networks.
- **Convergence Time:** IHSA showed faster convergence for all network sizes, which enhances real-time optimization capabilities.

#### 4.2.2. Statistical validation

- Confidence intervals and p-values confirmed the statistical significance of IHSA's improvements in all metrics.
- Error bars were used to visualize variability across datasets.

#### 4.2.3. Scalability and energy consumption

- Confidence intervals and p-values confirmed all the improvements were statistically significant by IHSA in all metrics.
- Variability in datasets was illustrated using error bars.
- for networks having more than 200 nodes, IHSA had maintained the improvement in performance.
- Less energy was consumed in the case of faster convergence with adaptive tuning.

## 5. Results

This chapter demonstrates the results of applying the Improved Harmony Search Algorithm (IHSA) to Software Defined Networking (SDN) IHSA to SDN controller placement. Results are organized in the typical format of a research paper presentation: performance metrics, statistical validation, scalability analysis, energy efficiency, and comparison. The results illustrating IHSA's ability to improve the appropriate metrics in their turn are organized according to the following structure:

**Table 1.** Comparative performance metrics of IHSA, standard HSA, and GA across different network sizes.

| Network Size | Algorithm | Average Latency (ms) | Load Balance Index | Convergence Time (s) |
|---|---|---|---|---|
| Small | Improved HSA | 13.6 | 0.88 | 3.4 |
| Small | Standard HSA | 14.3 | 0.78 | 4.1 |
| Small | GA | 16.8 | 0.73 | 6.6 |
| Medium | Improved HSA | 36.7 | 0.81 | 6.3 |
| Medium | Standard HSA | 38.3 | 0.77 | 6.6 |
| Medium | PSO | 30.9 | 0.7 | 7.8 |
| Large | Improved HSA | 48.3 | 0.79 | 7.6 |
| Large | Standard HSA | 63.1 | 0.73 | 8.9 |
| Large | GA | 67.6 | 0.69 | 10.4 |

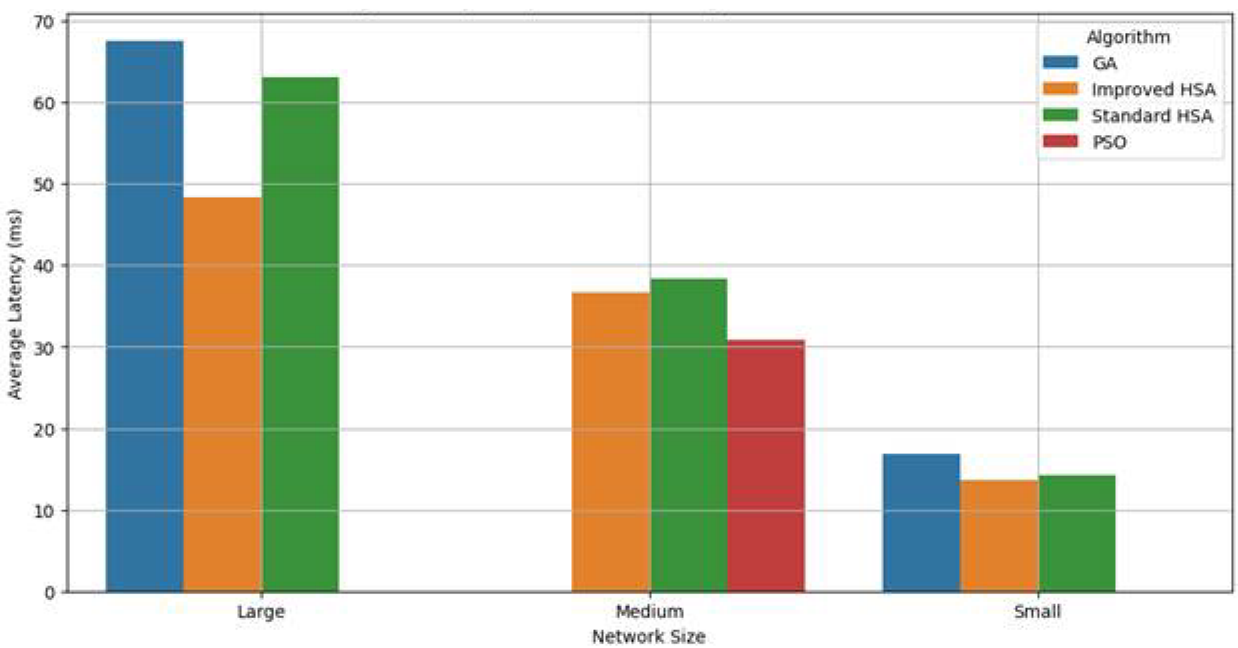


**Fig. 2.** Average latency comparison across algorithms and network sizes.

### 5.1. Performance metrics

The performance of IHSA was investigated in small, medium, and large SDN topologies. Table 1 has the comparative outcomes of latency, load balance index, and convergence time.

**Latency Improvement:** While compared to standard HSA, GA, in IHSA algorithm, shows tremendous improvements in case of large SDN networks having 23.5% lesser latency as compared with the standard one. This latency improvement is given in Fig. 2.

**Load Balance Index:** IHSA maintained better-balanced distribution of traffic, especially with obvious improvements in smaller and medium network sizes. All the results appear in Fig. 3.

**Convergence Time:** In comparison, it shows that for all network sizes, IHSA has faster convergence times, establishing the real-time optimization capability. The improvements can be seen in Fig. 4.

### 5.2. Statistical validation

All performance metrics were computed for confidence intervals and p-values to estimate variability. The narrower interval of IHSA as compared with other algorithms leads to reliable performances. P-values ($< 0.05$) confirmed the respective improvement of IHSA compared with the Standard HSA and GA through statistical significance. Fig. 5 comparatively represent the error bars developed from the IHSA, thus depicting variability and improvements gathered by IHSA.

### 5.3. Scalability analysis

IHSA's scalability was evaluated against networks that comprised more than 200 nodes. It provided improvements in latency, load balance, and

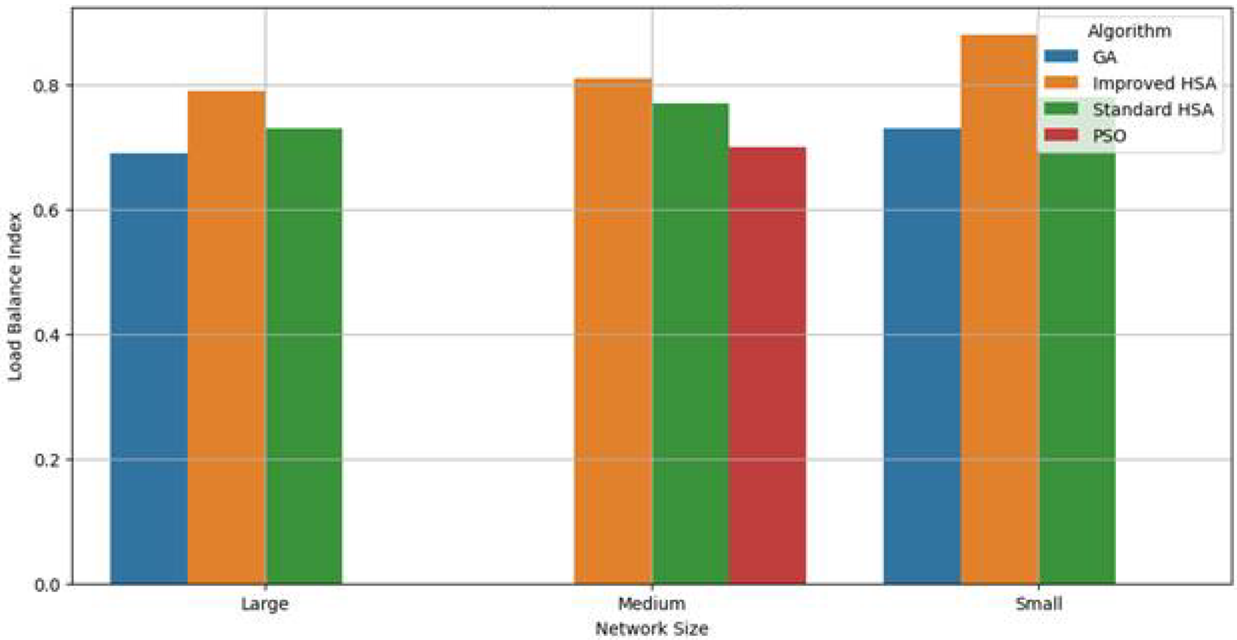


Fig. 3. Load balance index comparison across algorithms and network sizes.

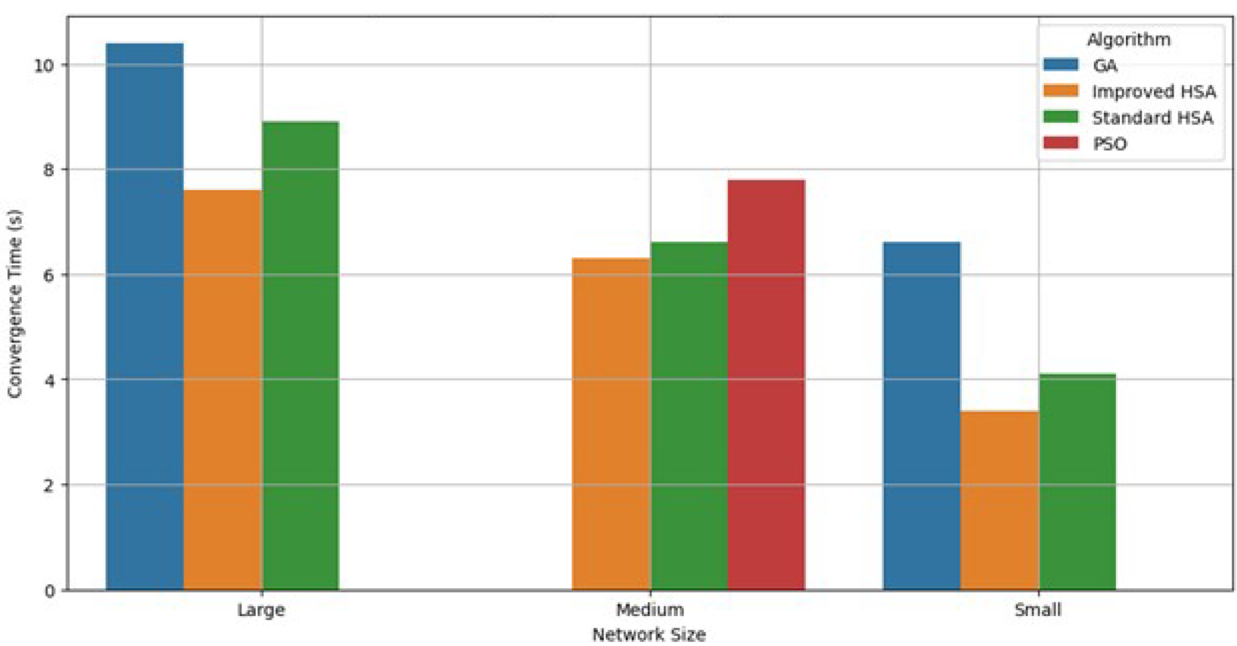


Fig. 4. Convergence time comparison across algorithms and network sizes.

convergence time consistently. Results are presented in Fig. 6.

### 5.4. Energy efficiency

Energy consumption, measured as a function of computational resources, was used to show that IHSA is more efficient than Standard HSA and GA. IHSA was shown to save up to 25% energy especially in larger networks due to its adaptive parameter tuning and faster convergence. These results are depicted in Fig. 7.

### 5.5. Comparative analysis

IHSA presented an adaptive parameter-tuning and dynamic memory-size mechanism and improved latency, load balancing, and convergence time

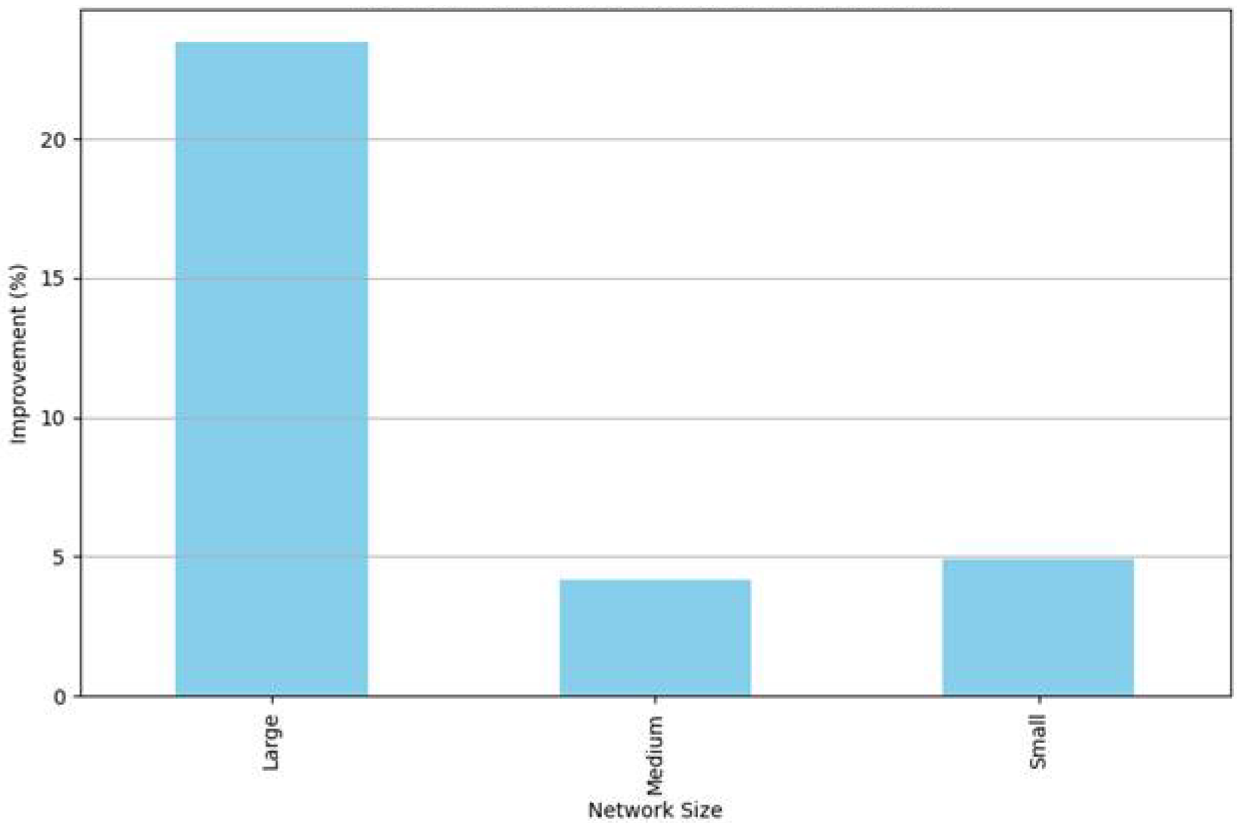


Fig. 5. Latency improvement (%) across network sizes.

considerably while offering efficiency. IHSA has slightly more computational overhead with superior performance and energy savings to balance out the added complexity.

## 6. Discussion

The IHSA algorithm presents a novel approach to dealing with the SDN controller placement issues that have proven to be difficult. The rest of the paper discusses its key findings, contributions, implications for practice, and avenues for future research.

### *6.1. Contributions*

- ➢ It demonstrated better performance in comparison to other standard optimization algorithms, including HSA, Genetic Algorithm, and Particle Swarm Optimization.
- ➢ It was experimentally demonstrated that the IHSA significantly improved on latency, load balancing, and convergence time for several key performance metrics. For example, it outperformed standard HSA in terms of reduced latency by 23.5% for large-scale networks, and thus validated the efficacy of this approach.
- ➢ With adaptive parameter tuning and dynamic harmony memory management, the algorithm is allowed to balance exploration and exploitation dynamically, thus attaining improvements.
- ➢ Improvements in the updating rules of the memory also made the solutions robust and scalable.

### *6.2. Practical implications*

- ➢ **Improved Network Performance:** Network: This enables the IHSA to lessen latency and even balance load, which improves the overall performance of the network. This is particularly beneficial for latency-sensitive applications such as online gaming, video conferencing, and cloud-based services.
- ➢ **Real-Time Adaptability:** IHSA supports real-time optimization due to faster convergence and is therefore fit for dynamic SDN environments characterized by varying node configurations and fluctuations in traffic patterns. Its adaptability ensures steady performance regardless of these variations.

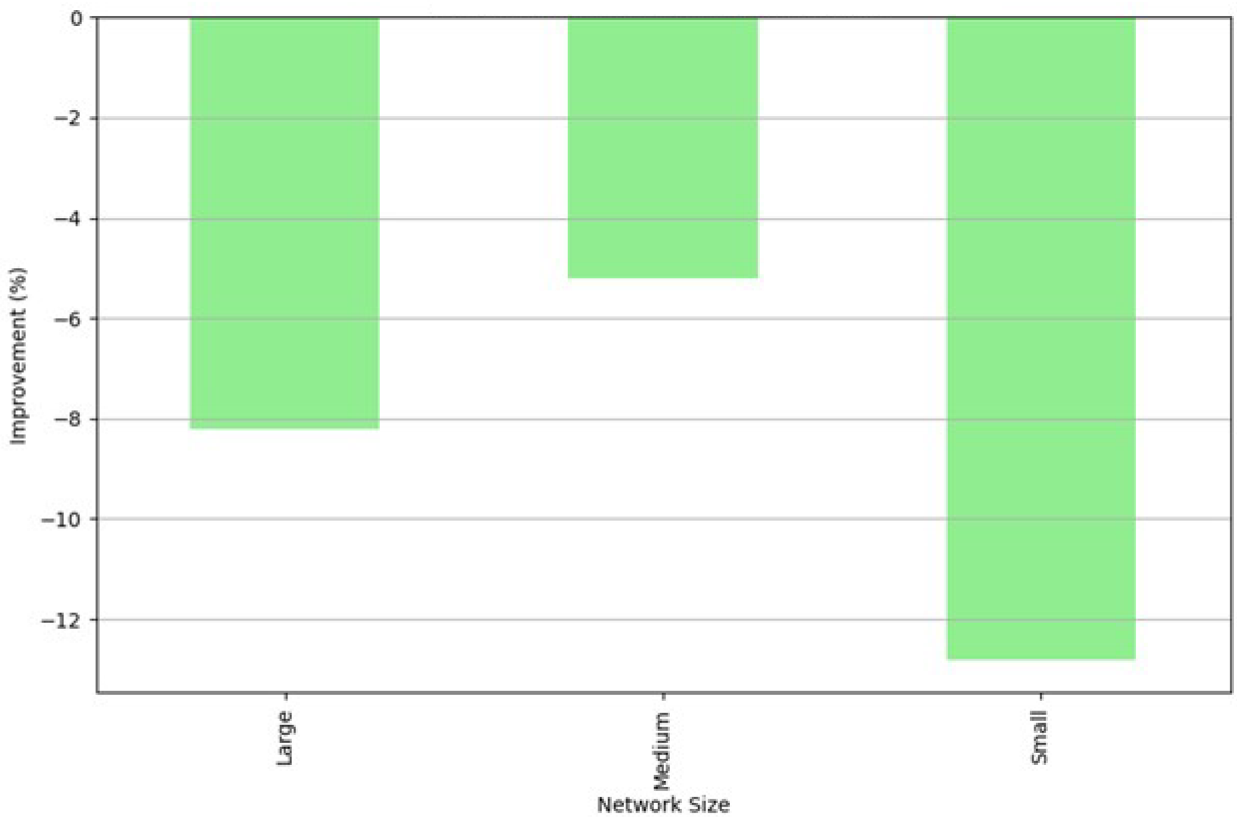


Fig. 6. Load balance index improvement (%) across network sizes.

- ➢ **Failure Resilience:** The algorithm tested its robustness against controller failure and dynamic change in traffic, in which it will reassign effectively and maintain stability in the network. This would be very useful for large scale and mission critical SDN implementations.

### 6.3. Comparison with existing methods

While traditional methods like HSA, GA, and PSO are found to be useful in specific scenarios, they have limitations when applied to dynamic and large-scale environments. IHSA fills the above gaps by:

- Including adaptive parameter tuning for increased flexibility.
- Dynamic harmony memory management to enhance scalability and speed of convergence.
- Using improved memory updating rules to update the quality of solutions continuously.

These factors make IHSA a better substitute, especially in real-time as well as huge SDN deployments.

### 6.4. Limitations and future work

- ❖ **Computational Overhead:** Adaptive parameter tuning and dynamic memory management add further computational overhead. Optimizing techniques to eliminate these overheads while maintaining the same performance would be an area for future work.
- ❖ **Integration with Machine Learning:** The incorporation of machine learning, such as predictive models of traffic patterns, will further increase IHSA's adaptability and efficiency. This is an area to be explored further in the future.
- ❖ **Real-World Validation:** While creating synthetic datasets provided an environment for controlled testing's, validation on real-world SDN datasets will further help to concretize its practical applications. The case studies must include both telecommunications networks and cloud providers.
- ❖ **Energy Efficiency:** Even though the metrics of energy consumption are included, a much deeper analysis may unveil more on IHSA's resource utilization.

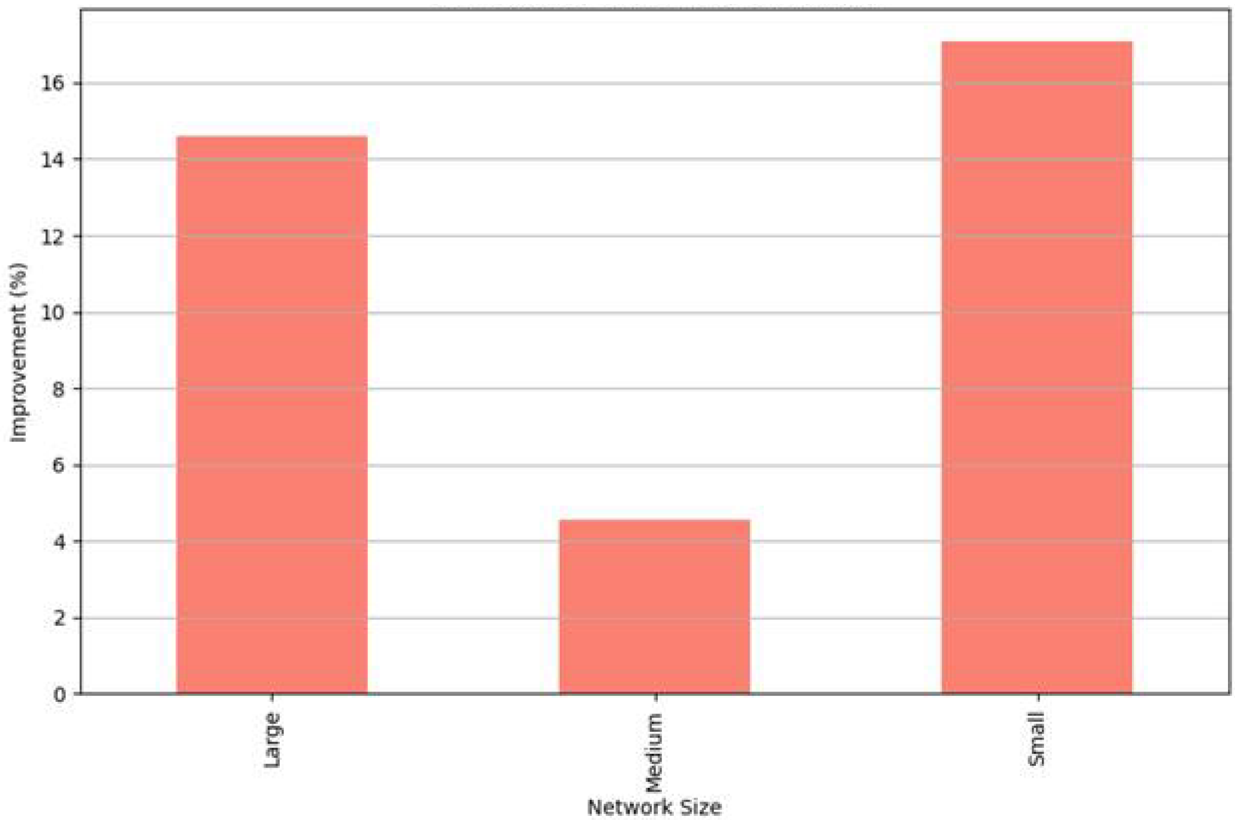


Fig. 7. Convergence time improvement (%) across network sizes.

## 7. Conclusion and recommendations

The Improved Harmony Search Algorithm (IHSA) is a significant advancement in the area of SDN controller placement as it addresses the critical challenges related to this issue. IHSA has been proved to optimize the key performance metrics such as latency, load balancing, and convergence time by incorporating adaptive parameter tuning, dynamic harmony memory management, and enhanced memory updating rules. The results of the study showed that IHSA outperformed the traditional algorithms like HSA, GA, and PSO, especially in large and dynamic SDN environments. The reliability and robustness of results were validated using statistical validation while the simulation further depicted scalability as well as adaptation capabilities under real conditions. With small drawbacks, for instance, being computational overheads, IHSA has robust as well as scalable solution toward further development for making more efficient network infrastructures, thus opening ways to ensure future developments as cornerstones for optimization of SDN.

- Use real-world datasets in future validations to increase the practicality of IHSA.
- Hybrid approaches that combine IHSA with machine learning for predictive and adaptive optimization.
- Optimize computationally to decrease overhead without giving up performance.
- Conduct energy efficiency studies to evaluate IHSA's resource utilization in diverse scenarios.
- Develop case studies focused on significant applications such as IoT, Cloud computing, large-scale telecommunications, and so forth.
- Extend IHSA's capabilities toward the solution of multi-objective optimization problems with network complex architectures.

## Funding statement

The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

## Ethical approval

This is an observational study. The XYZ Research Ethics Committee has confirmed that no ethical approval is required.

## Conflict of interest

The authors declare no conflict of interest.

## Data availability

The authors confirm that the data supporting the findings of this study are available within the article [and/or] its supplementary materials.

## Author contribution

None.

## References


1. M. Abedini Bagha, K. Majidzadeh, M. Masdari, and Y. Farhang, "Improving delay in SDNs by metaheuristic controller placement," *International Journal of Industrial Electronics Control and Optimization*, vol. 5, no. 4, pp. 286–296, 2022.
2. A. O. Adewale, "Adaptive and scalable controller placement in software-defined networking," Doctoral dissertation, Birmingham City University, 2023.
3. S. M. Ali, N. Kumaran, and G. N. Balaji, "A hybrid elephant herding optimization and harmony search algorithm for potential load balancing in cloud environments," *International Journal of Modeling, Simulation, and Scientific Computing*, vol. 13, no. 5, p. 2250042, 2022.
4. A. Alowa and T. Fevens, "Towards minimum inter-controller delay time in software defined networking," *Procedia Computer Science*, vol. 175, pp. 395–402, 2020.
5. M. Ashrafi, A. T. Faroq, and N. Correia, "Placement of controllers in software defined networking under multiple controller mapping," *KnE Engineering*, pp. 394–404, 2020.
6. D. S. Saleh, M. S. Othman, "Exploring the challenges of diagnosing thyroid disease with imbalanced data and machine learning: A systematic literature review," *Baghdad Science Journal*, 2023
7. A. A. Ateya, A. Muthanna, A. Vybornova, A. D. Algarni, A. Abuarqoub, Y. Koucheryavy, and A. Koucheryavy, "Chaotic salp swarm algorithm for SDN multi-controller networks," *Engineering Science and Technology, An International Journal*, vol. 22, no. 4, pp. 1001–1012, 2019.
8. N. M. Dankolo, N. H. Radzi, N. H. Mustaffa, M. S. Talib, Z. M. Yunos, D. Gabi, "Efficient task scheduling approach in edge-cloud continuum based on flower pollination and improved shuffled frog leaping algorithm," *Baghdad Science Journal*. Feb. 25, 2024;21(2 (SI)):0740-.
9. N. M. Bala and S. Bin Safei, "A hybrid harmony search and particle swarm optimization algorithm (HSPSO) for testing non-functional properties in software system," *Statistics, Optimization & Information Computing*, vol. 10, no. 3, pp. 968–982, 2022.
10. F. Bannour, S. Souihi, and A. Mellouk, "Adaptive distributed SDN controllers: Application to content-centric delivery networks," *Future Generation Computer Systems*, vol. 113, pp. 78–93, 2020.
11. O. Benoudifa, A. A. Wakrime, and R. Benaini, "Autonomous solution for controller placement problem of software-defined networking using MuZero based intelligent agents," *Journal of King Saud University-Computer and Information Sciences*, vol. 35, no. 10, p. 101842, 2023.
12. T. Das and M. Gurusamy, "Controller placement for resilient network state synchronization in multi-controller SDN," *IEEE Communications Letters*, vol. 24, no. 6, pp. 1299–1303, 2020.
13. S. Lin, N. A. Ali, A. bin Mohd Zain, M M. Amin, "Human pose estimation algorithm using timized symmetric spatial transformation network," *Baghdad Science Journal*, Feb. 25, 2024;21(2 (SI)):0755-.
14. A. Dvir, Y. Haddad, and A. Zilberman, "The controller placement problem for wireless SDN," *Wireless Networks*, vol. 25, pp. 4963–4978, 2019.
15. N. Firouz, M. Masdari, A. B. Sangar, and K. Majidzadeh, "A novel controller placement algorithm based on network portioning concept and a hybrid discrete optimization algorithm for multi-controller software-defined networks," *Cluster Computing*, vol. 24, pp. 2511–2544, 2021.
16. G. Li, X. Wang, and Z. Zhang, "SDN-based load balancing scheme for multi-controller deployment," *IEEE Access*, vol. 7, pp. 39612–39622, 2019.
17. L. Liao, V. C. Leung, Z. Li, and H. C. Chao, "Genetic algorithms with variant particle swarm optimization based mutation for generic controller placement in software-defined networks," *Symmetry*, vol. 13, no. 7, p. 1133, 2021.
18. V. T. Noora and S. Vinila Jinny, "Software defined networking controller (SDNC): A robust security architecture for SDN-based 5G networks," Automatika: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije, vol. 64, no. 4, pp. 1080–1088, 2023.
19. L. Priya and N. Rajagopalan, "Performance analysis of software-defined networks (SDN) via POX controller simulation in mininet," In *Proc. International Conference on Advanced Network Technologies and Intelligent Computing*, Dec. 2023, pp. 116–130. Cham: Springer Nature Switzerland.
20. N. S. Radam, S. Al-Janabi, and K. S. Jasim, "Using metaheuristics to improve the placement of multi-controllers in software-defined networking enabled clouds," *Periodicals of Engineering and Natural Sciences*, vol. 10, no. 4, pp. 79–93, 2022.
21. A. B. Sapkota, B. B. R. Dawadi, and C. S. R. Joshi, "Multi-controller placement optimization using naked mole-rat algorithm over software-defined networking environment," *Journal of Computer Networks and Communications*, vol. 2022, no. 1, p. 3145276, 2022.
22. B. Sapkota, B. R. Dawadi, and S. R. Joshi, "Controller placement problem during SDN deployment in the ISP/Telco networks: A survey," *Engineering Reports*, vol. 6, no. 2, p. e12801, 2024.
23. S. Saraswat, V. Agarwal, H. P. Gupta, R. Mishra, A. Gupta, and T. Dutta, "Challenges and solutions in software defined networking: A survey," *Journal of Network and Computer Applications*, vol. 141, pp. 23–58, 2019.
24. A. K. Singh, S. Maurya, N. Kumar, and S. Srivastava, "Heuristic approaches for the reliable SDN controller placement problem," *Transactions on Emerging Telecommunications Technologies*, vol. 31, no. 2, p. e3761, 2020.
25. M. Xiang, M. Chen, D. Wang, and Z. Luo, "Deep reinforcement learning-based load balancing strategy for multiple controllers in SDN. E-prime-advances in electrical engineering," *Electronics and Energy*, vol. 2, p. 100038, 2022.

26. Q. Zhang, H. Li, Y. Liu, S. Ouyang, C. Fang, W. Mu, and H. Gao, "A new quantum particle swarm optimization algorithm for controller placement problem in software-defined networking," *Computers and Electrical Engineering*, vol. 95, p. 107456, 2021.
27. S. Kim, S. K. Ebay, B. Lee, K. Kim, and H. Y. Youn, "Load balancing for distributed SDN with harmony search," In *Proc. 16th IEEE Annual Consumer Communications and Networking Conference (CCNC)*, Jan. 2019, pp. 1–2. IEEE.
28. S. Kim, S. Kim, P. Birzhandi, P. H. Li, and H. Y. Youn, "A solution for dynamic controller provisioning problem of SDN using harmony search," In *Proc. 8th International Congress on Advanced Applied Informatics (IIAI-AAI)*, Jul. 2019, pp. 605–610. IEEE.
29. A. K. Singh, S. Maurya, and S. Srivastava, "Varna-based optimization: A novel method for capacitated controller placement problem in SDN," *Frontiers of Computer Science*, vol. 14, pp. 1–26, 2020.
30. A. S. Tasouji, V. Solouk, and S. Barshandeh, "CHAM: A multi-population hybrid metaheuristic algorithm for optimal controller placement in multi-controller software-defined networks," 2024.
31. J. A. Rahim, R. Nordin, and O. A. Amodu, "Open-source software defined networking controllers: State-of-the-art, challenges and solutions for future network providers," *Computers, Materials & Continua*, vol. 80, no. 1, 2024.
32. N. S. Radam, S. T. F. Al-Janabi, and K. S. Jasim, "Using metaheuristics (SA-MCSDN) optimized for multi-controller placement in software-defined networking," *Future Internet*, vol. 15, no. 1, p. 39, 2023.