Computers, Materials & Continua

Tech Science Press

DOI: 10.32604/cmc.2023.037141
*Article*

# Sea Turtle Foraging Optimization-Based Controller Placement with Blockchain-Assisted Intrusion Detection in Software-Defined Networks

**Sultan Alkhliwi***

Department of Computer Science, Faculty of Science, Northern Border University, Arar, Saudi Arabia

*Corresponding Author: Sultan Alkhliwi. Email: salkhliwi@nbu.edu.sa
Received: 25 October 2022; Accepted: 17 February 2023

**Abstract:** Software-defined networking (SDN) algorithms are gaining increasing interest and are making networks flexible and agile. The basic idea of SDN is to move the control planes to more than one server's named controllers and limit the data planes to numerous sending network components, enabling flexible and dynamic network management. A distinctive characteristic of SDN is that it can logically centralize the control plane by utilizing many physical controllers. The deployment of the controller—that is, the controller placement problem (CPP)—becomes a vital model challenge. Through the advancements of blockchain technology, data integrity between nodes can be enhanced with no requirement for a trusted third party. Using the latest developments in blockchain technology, this article designs a novel sea turtle foraging optimization algorithm for the controller placement problem (STFOA-CPP) with blockchain-based intrusion detection in an SDN environment. The major intention of the STFOA-CPP technique is the maximization of lifetime, network connectivity, and load balancing with the minimization of latency. In addition, the STFOA-CPP technique is based on the sea turtles' food-searching characteristics of tracking the odour path of dimethyl sulphide (DMS) released from food sources. Moreover, the presented STFOA-CPP technique can adapt with the controller's count mandated and the shift to controller mapping to variable network traffic. Finally, the blockchain can inspect the data integrity, determine significantly malicious input, and improve the robust nature of developing a trust relationship between several nodes in the SDN. To demonstrate the improved performance of the STFOA-CPP algorithm, a wide-ranging experimental analysis was carried out. The extensive comparison study highlighted the improved outcomes of the STFOA-CPP technique over other recent approaches.

**Keywords:** Software-defined networking; NP hard problem; metaheuristics; controller placement problem; objective function

## 1 Introduction

The current advancements in information and communication technology (ICT), namely traffic growth, cloud computing (CC), video conferencing, social networks, the Internet of Things (IoT), and online gaming, demonstrate that conventional networks do not have the capability to meet the requirements of traffic-based to novel applications and network management necessities were confronting important challenges [1]. Owing to the proliferation of networks, the complexity of networks and the number of elements seems to be escalating; thus, the quality of service (QoS) and network management provision for novel applications faces significant difficulties [2]. As per the wireless network resource, there arise various kinds of necessities that certain research works paid to such complexities such as energy consumption and QoS, which is mentioned above. Thus, it becomes essential to provide, upgrade, and manage innovative services without including novel hardware in the next-generation network [3]. In recent times, the segregation of control planes from the data planes in Software-defined networking (SDN) and a central controller that has a universal view of the network have been formulated for improving and managing network resource management in next generation networks (NGN) [4]. Fig. 1 defines the process of SDN.

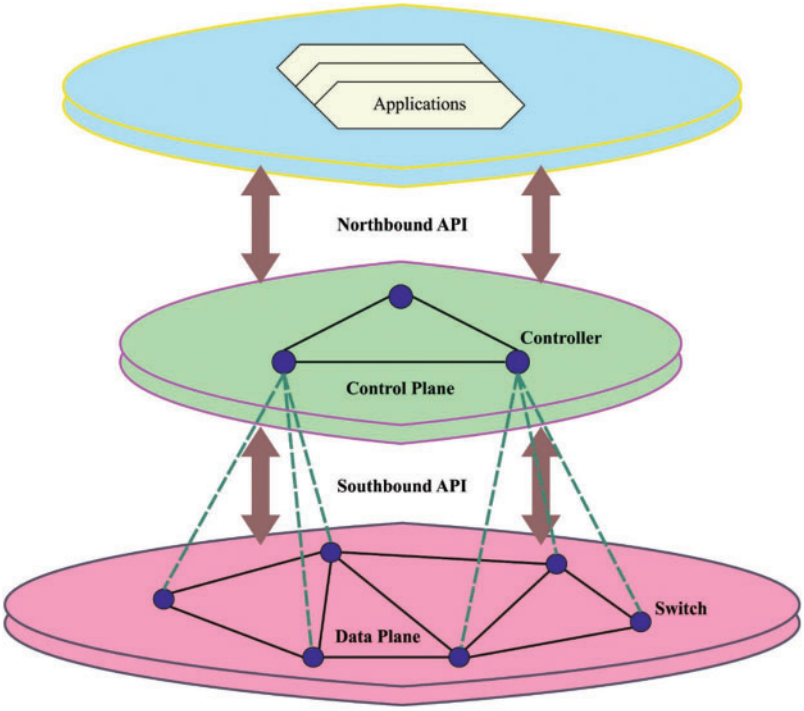


**Figure 1:** Process of SDN

The representative southbound interfaces of SDN can be OpenFlow, which will primarily assume that there will be just one controller in the network for simplicity [5]. It becomes difficult for one controller to meet the wide management requirements with the expansion of the SDN network. In contrast, in SDN-related network systems, controllers have the main role in handling network traffic and enhancing network resource efficiency [6]. To enhance the reliability and scalability of the network and prevent a single point of failure, logically centralized, physically distributed multi-control network structures emerge, namely Onix, HyperFlow, and Kandoo, among others. Therefore, the failure of the controller or controllers could affect network performance. Hence, the number of switches of

controllers rises, and this progression can cause further complexities. Whenever a single point of failure begins to occur in a controller, it will induce problems in controlling multiple flow routing in the Wide Area Network (WAN) [7]. In contrast, a single controller could turn out to be a bottleneck in various dimensions—that is, bandwidth, processing, the number of input/output (I/O), and Random Access Memory (RAM) will cause a rise in propagation delay among functional electrical stimulation (FEs) and controllers. Therefore, the use of many controllers in WANs is unavoidable for maintaining reliability, efficiency, scalability, and diminishing the propagation delay among controllers and FEs.

As the location and several controller deployments have a massive effect on the performance of the network in a multi-controller network structure, the "controller placement problem" (CPP) is becoming a hotspot in present SDN research [8]. For a specified network, CPP mainly has three problems to resolve. The first is the location of the controllers. The second is the allotment between controller and switch, aiming to maximize performance variables such as shortening the latency, raising the energy efficiency, improving the reliability, and so on. Finally, CPP addresses the number of controllers required [9]. The problem includes identifying the best controllers, location optimization, and finding the set of switches that are handled by every controller. The issue of controller placement is relevant even for a network having a single physical controller; however, the problem is less pronounced [10]. The first researchers to study CPP were Heller et al., who developed the issue as a facility location problem and considered it as NP-hard. Since there were several efforts for placing the controllers optimally.

This article designs a novel sea turtle foraging optimization algorithm for the controller placement problem (STFOA-CPP) in an SDN. The STFOA-CPP technique mainly relies on the sea turtles' food-searching characteristics of tracking the odour path of dimethyl sulphide (DMS) released from its food sources. Furthermore, the presented STFOA-CPP technique can adapt to the number of controllers required and the switch to controller mapping to variable network traffic. Finally, the blockchain can inspect data integrity, determine significantly malicious input, and improve the robust nature of developing a trust relationship between several nodes in the SDN. For assuring the enhanced outcomes of the STFOA-CPP algorithm, a series of simulations were executed.

## 2 Literature Review

The author in [11] developed a fault-tolerance metaheuristics-related technique for CPP in a wireless SDN, called fault-tolerance metaheuristic-based scheme (FTMBS). The objectives of the technique are to diminish the worst-case latency among associated nodes and controllers, maximize network connectivity, maximize lifetime of the network, and maximize the load balance amongst controllers. When handling a multi-objective system, a trade-off exists when these multi-objective metrics compete with one another, and the decision maker decides on these trade-offs. Wu et al. [12] introduced a deep Q-network (DQN) enabled dynamic flow data-driven methodology for the controller placement problem (D4CPP). D4CPP incorporates past network data learning into the real-time switch-controller mapping decision and controller deployment, to be adopted into the dynamic network environment with flow fluctuation. Specifically, D4CPP considered load balance, data latency, and flow fluctuation, and could accomplish an enhanced balance between these metrics.

The authors in [13] examined the CPP for multiple link failures (CPP-MLF). Initially, real link information of the present network was examined to formalize the connection failure rate and examine the features of link failures. Then, minimalizing the worst-case delay and the number of controllers as the optimization objective, a heuristic technique was developed based on enhanced non dominated sorting genetic algorithm II (NSGA-II) to efficiently resolve the presented algorithm. Li et al. [14]

established a parameter optimization model (POM) using a heuristic technique employed in the CPP. The heuristic approach could efficiently resolve the CPP via the optimized parameter attained in the POM. In order to authenticate the efficiency of the POM, the authors first established a synthetically delayed CPP technique for reducing the delay among switches as well as controllers.

In [15], a novel multi-objective version of the marine predator algorithm (MOMPA) was proposed. This was then hybridized with non-dominated sorting genetic algorithm II (GA-II). The presented technique can be discretized with crossover and mutation functions. Subsequently, the presented hybrid discrete multi-objective model has been applied to resolve the CPP. In [16], a new CPP that considers control plane structure and relationships among data and control planes was designed. This architecture is regarded as a multi-objective optimization technique with two objective functions to reduce inter-controller latency and flow setup time. As a result, we adapted the best–worst multi-criteria decision-making technique, which considered three metrics—link utilization, hop count, and propagation latency—to allocate switches to the controller.

Kotachi et al. [17] developed a CPP technique that permits distribution against SDN controller failure. The presented technique defines the ratios of computation ability required by the SDN switches on the SDN controller linked to it. Furthermore, the presented technique defines the allocation of SDN switches to the SDN controller, the placement, and the number of SDN controllers. The CPP can be defined such that a network provider continues to lead each SDN switch if no more than a specific amount of SDN controller failure occurs. From this, two load-distribution techniques are devised: split and even-split. Guan et al. [18] developed the idea of a synthetic delay and took the controller ability as constrained conditions to establish a CPP technique. The feature selection (FA) was selected as the problem-solving technique for the CPP. As a result, we initially developed the dynamic parameter approach for controlling parameters of the random walk of FA, the light absorption co-efficient, and the light attractiveness. Next, the particle swarm optimization (PSO) technique is adopted to enhance the static parameter of the FA.

## 3 The Proposed Model

In this study, a new STFOA-CCP algorithm was devised for CPP in the SDN environment. The goal of the STFOA-CPP technique is to maximize lifetime, network connectivity, and load balancing with minimal latency. Moreover, the presented STFOA-CPP technique can adapt to the number of controllers mandated and the switch to controller mapping for variable network traffic. Finally, the blockchain can inspect data integrity, determine significantly malicious input, and improve the robust nature of developing a trust relationship between several nodes in the SDN.

### *3.1 Algorithmic Design of STFOA*

The STFOA algorithm is inspired by the hunting behaviour of a sea turtle. This section, therefore, provides certain detail regarding the life of sea turtles. There are seven species of sea turtle in two families. Six species are in the *Cheloniidae* family; however, the leatherback belongs to the *Dermochelyidae* family [19]. Sea turtles are considered to be skilled ocean navigators. They can travel millions of kilometres in their lifetime, nesting and feeding. The sea turtle usually travels in open waters, wherein the current will affect its movement. It will be moving linearly from one point to another. Sea turtles might drift inactively with the ocean currents or swim determinedly towards a fixed point. At the time of their long-distance migration, sea turtles feed on jellyfish and other invertebrates that seem to be copious in frontal regions of the ocean, consisting of phytoplankton in high concentrations. When it is fed upon, phytoplankton releases a substance called dimethylsulfoniopropionate (DMSP),

which can break down into dimethyl sulphide (DMS), an odorous chemical. As DMS is volatile, it can be detected in the air beyond an ocean area that contains phytoplankton in abundance. DMS can serve as a pointer for a high density of prey for sea turtles. In real-time, sea turtles detect DMS, using this sensing capability to find favourable foraging regions.

This research approach is modelled after the food-searching technique of sea turtles. They sense the odour of DMS originated from their food source and move towards the food source that discharges the strong odour. Furthermore, the turtle's movement can be assisted by ocean currents. The presented technique, which is inspired by the abovementioned foraging process, is defined as follows:

Step 1: Specify a population of $N$ turtles and arbitrarily determine the initial location of every turtle [20]:

$$P_i(0) = \left[p_1^i, p_2^i, \ldots p_D^i\right], \tag{1}$$

where $i = 1$ to $N$ within a $D$ dimension continuous search space.

Step 2: Randomly produce the initial velocity of sea turtles, $V_i(0) = [v_1^i, v_2^i, \ldots v_D^i]$. The velocity of turtles is constrained to be within the limit characterized in the following equation:

$$V_{\max} = \alpha\,(XUB - XLB)\,, \tag{2}$$

$$V_{\min} = -V_{\max}, \tag{3}$$

where $XUB$ and $XLB$ denote the upper and lower limits of the $D$ dimension search space, and $\alpha$ represents a constant within [0, 1].

Step 3: Randomly produce the initial position of $M$ food source:

$$K_j(0) = \left[k_1^j, k_2^j, \ldots, k_D^j\right], \tag{4}$$

where $j = 1$ to $M$, within a $D$ dimension continuous search space.

Step 4: Input the location of every food source into the objective function and estimate them to accomplish the fitness values of that food source.

Step 5: Input the location of every turtle into the objective function and evaluate them to achieve the fitness values; the one with the maximum fitness value, $I$, is recorded.

$$I = \arg\ \max_i \left(f_{P_i}(t)\right), \tag{5}$$

where $f_{P_i(t)}$ denotes the fitness value of turtle $i$ at time $t$.

Step 6: Upgrade the velocity of every sea turtle using the following:

$$V_i(t) = V_i(t-1) + \left(\frac{f_{P_i}(t) - f_{P_i}(t-1)}{f_{P_i}(t-1)}\right)\left(P_i(t) - P_i(t-1)\right), \tag{6}$$

where $P(t)$ refers to the location of the turtle at time $t$ and $P_i(t-1)$ represents its location at time t – 1.

Step 7: Evaluate the velocity of the ocean current, $VC_i(t) = \left[vc_1^i, vc_2^i, \ldots vc_D^i\right]$:

$$VC_i(t) = \alpha\left(P_I(t) - P_i(t)\right). \tag{7}$$

Step 8: Add the velocity of the turtle to the ocean current to obtain the combined velocity:

$$VM_i(t) = V_i(t) + VC_i(t)\,. \tag{8}$$

Step 9: Compare the fitness values of every turtle with the fitness values of every food source. When the fitness value of the turtle is greater than the food source, the contribution of the food source

becomes zero. However, when the fitness value of the turtle is less than that of the food sources, the contribution of food sources is set as follows:

$$CK_j = \frac{f_{K_j}}{\sum_{q=1}^{M} f_{K_q}}, \tag{9}$$

where $f_{K_j}$ indicates the fitness value of food source $j$.

Step 10: Evaluate the distance between the food source and the turtle:

$$d_{ij} = \left\| P_i - K_j \right\|. \tag{10}$$

Step 11: Evaluate the level of DMS odour from food source $j$ perceived by the turtle, $C_{ij}(t)$. The level of DMS odour affects the movement of the turtle to its location:

$$C_{ij}(t) = \left( CK_j \times \exp\left( -\frac{d_{ij}^2}{2\sigma^2(t)} \right) \right), \tag{11}$$

$$\sigma(t) = \sigma_0 \exp\left( -\frac{t}{T} \right). \tag{12}$$

From the expression, $\sigma(t)$ denotes the fading of the level of DMS odour with passing time, $\sigma_0$ denotes a constant equivalent to 1, and $T$ shows the number of iterations after which the odor completely disappears.

Step 12: Define better food sources for the turtle $i$. The optimal food source is the one that has the maximum value of $C(t)$ among each food source.

$$J = \arg\ \max_j (C_{ij}) j. \tag{13}$$

Step 13: Upgrade the location of every turtle based on the following expression:

$$P_i(t+1) = P_i(t) + \eta VM_i(t) + C_{iJ}(t)(K_J - P_i). \tag{14}$$

Step 14: Check the termination condition. When the maximal number of iterations is accomplished, the process ends. Otherwise, the succeeding two criteria are checked: (i) when the value of t/T is an integer, return to step 3; (ii) when the value of t/T is not an integer, return to step 5.

### *3.2 Process Involved in STFOA-CPP Technique*

Consider the SDN as an undirected graph $G(V, E)$ with vertices $|V|$ as a number of links $E$ and sensor nodes. It consists of a number of controllers $C_n$, $n = 1, 2, 3$; one fault-free executed at a sink termed $C_{root}$ and the remainder to be chosen from cluster heads (CHs), $\{CH_1, CH_2, \ldots, CH\}$ using a metaheuristic algorithm. The system constraint is given below:

- Maximizing the flow amongst the controllers. The flow can be described by the average connectivity, which shows the network's reliability. It is the sum of each flow message received at the $C_i$ controller distributed from CH, signified as $fCH_i, C_i$. It should be noted that when a disjoint path exists in a faulty node, the message could still reach its terminus because the forwarding table includes alternate paths to the controller. Maximizing the average connectivity might result in the maximization in global network consistency, and this is shown in the following expression:

$$f1 = \sum_{i=1}^{\#CH \epsilon Ci} \frac{fCH_i, C_i}{\#CH}. \tag{15}$$

- Maximizing the load balance amongst controllers. A significant factor when applying multiple controllers to load balancing between the least loaded and the heaviest loaded controller is preventing overwhelming of the controllers, which adversely impacts controller response time and generates redundancy delays. This becomes a crucial problem that must be taken into account, particularly in the in-bound system, and it is expressed in the following equation:

$$f2 = \text{Max}\ \min_k\ (loadd C_i T_n), \tag{16}$$

where *load* $C_i T_n$ indicates the load of $i$ controller represented as $C_i$ in chromosome $k$ at time $T_n$, which is the sum of control plane load induced by $CH$related to $C_i$, indicated as $fCH_i T_n$, plus the sum of control plane load induced by $CH$of the adjacent controller $j$ at time $T_n$, indicated as $fCH_j T_n, j \neq i$; consequently, it is shown below:

$$Load\ C_i T_n = \sum\nolimits_{CH_i \epsilon C_i} fCH_i T_n + \sum\nolimits_{CH_j \epsilon C_j} fCH_j T_n. \tag{17}$$

- Minimizing the worst-case latency indicated as $D_T$: $D_T$ represents the sum of the end-to-end delay of $C_i$ and $H_i$, and the next delay is the queuing delay of the controller in chromosome $k$, namely, the average duration expended in the controller systems. Controllers are modelled as $M/M/1$[37]; $\mu$ indicates service rates, and $\lambda$ denotes arrival rates. Therefore, the final aim was to limit the overall latency to the provided threshold: $D_T <= T_{threshold}$ as follows:

$$f3 = D_T = \min\left(\max_k\left(\sum^{no.CHs} \frac{distance\,(CH_i, C_i)}{1ight_{speed}} + \sum\nolimits_K \frac{1}{\mu - \lambda}\right)\right). \tag{18}$$

$$s.t D_T <= T_{threshold}$$

- Maximizing the lifetime of the network. This can be described as the duration required for the last node to die, where the node decreases when its residual energy (RE), represented as $\gamma_i$, falls under a certain energy threshold indicated by $E_{th}$. The network lifetime equation is shown in (19). Faulty nodes, represented as $f_{node}$, are eliminated by considering if a path represented by $p$ occurs among a non-faulty CH and controller on one side and the link efficacy, represented by $L_{eff}$, of these paths that are bounded to a threshold represented as $L_{effthreshold}$ on the other side.

$$f4 = \max_k \sum \gamma_i \times \sum L_{eff} \times p \times f_{node} \tag{19}$$

$$s.t.\text{ma}_{sk}\,(\gamma_i) < E_{th}, L_{eff} \leq L_{effthreshold}$$

This objective function raises energy efficiency and fault tolerance. Consider the RE, and link efficacy is represented by Eq. (20), consider the energy consumed for communication represented as $E_N$ at $d$ distance. The path condition between the controller and cluster head is shown below:

$$L_{eff} = \frac{B \log_2 (1 + SNR)}{EN} \times d. \tag{20}$$

Similarly, the system considers whether the CH nodes are faulty or not; namely, it removes the faulty node for controller placement.

$$p = \begin{cases} 1000 & if\ path\ exists\ between\ CH\ and\ controller \\ 1 & otherwise \end{cases} \tag{21}$$

$$fnode = \begin{cases} -1 & if\ CH\ is\ faulty \\ 1 & otherwise \end{cases} \tag{22}$$

We adopt the scalarization technique to resolve the multi-objective fitness function that generates one solution, and the weight is defined beforehand using the optimization technique.

$$F = \max\left(\omega_1 f1 + \omega_2 f2 + \omega_3 f3 + \omega_4 f4\right) \tag{23}$$

$$s.t \omega_1 + \omega_2 + \omega_3 + \omega_4 = 1.$$

- The *ROC* weight is formulated as follows, with $n = 4$; thus, it contains four functions.

$$\omega i = 1/_n \sum_{k=i}^{n} 1/k. \tag{24}$$

where $\omega_1 = 25/48$, $\omega_2 = 13/48$, $\omega_3 = 7/48$, and $\omega_4 = 3/48$. For example, consider the maximum weight value $f1$ that characterizes the average network connectivity.

### *3.3 Blockchain Enabled Intrusion Detection*

A blockchain is a distributed ledger technology that allows information to be globally distributed and stored on dissimilar servers and nodes that are shared openly in an unchanged record of transactions. A blockchain (electronic ledger) primarily involves a chronological order using discrete timestamps and a list of digital records (blocks) [21]. A digital record comprises numerous items, including cryptographic, payload, and timestamp values. The genesis block is the initial record in a blockchain, and the subsequent block may be linked to the preceding one via cryptographic hashing, making it an auditable and verifiable record. Furthermore, a blockchain can be upgraded through consensus algorithms amongst each party in the network, and when novel information is added, it remains unchanged. Every SDN node can interconnect with the others and interchange essential information or data. A node might have numerous components, including collaboration components (for exchanging data), connection components (for physical connection), trust management components (to measure node reputation), and blockchain components (to communicate with the chain). Both anomaly-based and rule-based detection techniques may be deployed in the node based on the requirements. A blockchain is established and updated through the consensus protocols and smart contracts agreed upon between each intrusion detection system (IDS) node. The consensus is expanded to the SDN controller and application. According to the scheme and requirements, different data may be chained, for example, rules, messages, and alarms. The blockchain guarantees the integrity of the data and facilitates the data being visible to others. For instance, each SDN plane could access the chain to retrieve predictable data. In real-time usage, a privacy-preserving technique is used for protecting privacy. SDN and blockchain can work and complement one another; namely, trust management is improved by retrieving data from the chain, and the SDN controller can enforce the policy. The architecture can preserve the advantages of trust-based IDS, SDN, and blockchain.

## 4 Experimental Validation

The experimental validation of the STFOA-CPP model is tested under two topologies: the Internet2 topology, with few nodes and few links (34 nodes and 42 links), and the PlanetlabV2 topology, with few nodes and a large number of links (41 nodes and 544 links). Table 1 offers a detailed CP cost analysis of the STFOA-CPP method based on the Internet2 topology with five controllers [6]. Finally, the performance of STFOA-CPP is compared with biogeography-based optimization (BBO)

[22], biology and behaviour (BAT) algorithm [23], framework of fireworks algorithm (FWA) [24] and improved quantum-behavior particle swarm optimization algorithm (FE-QPSO) [25].

**Table 1:** CP cost analysis of STFOA-CPP approach based on Internet2 topology with five controllers

| Cost: Internet2 topology, controllers = 5 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 0 | 3016848 | 3015289 | 2964634 | 3015289 | 2950606 |
| 100 | 3011393 | 2985675 | 2960737 | 2987234 | 2945930 |
| 200 | 3005938 | 2980220 | 2961517 | 2969310 | 2942813 |
| 300 | 3007496 | 2979441 | 2958399 | 2965413 | 2939696 |
| 400 | 3003600 | 2977103 | 2956841 | 2959958 | 2938137 |
| 500 | 2998924 | 2977103 | 2959958 | 2956841 | 2934240 |
| 600 | 2998924 | 2977103 | 2957620 | 2954503 | 2932682 |
| 700 | 2998924 | 2975544 | 2959179 | 2948268 | 2931123 |
| 800 | 2998144 | 2975544 | 2963855 | 2941254 | 2928785 |
| 900 | 2995807 | 2971648 | 2961517 | 2941254 | 2928785 |
| 1000 | 2992689 | 2971648 | 2960737 | 2942813 | 2928785 |
| 1100 | 2994248 | 2970089 | 2960737 | 2939696 | 2928785 |
| 1200 | 2994248 | 2970089 | 2958399 | 2938916 | 2925668 |
| 1300 | 2991131 | 2968530 | 2960737 | 2938137 | 2925668 |
| 1400 | 2989572 | 2966972 | 2960737 | 2933461 | 2923330 |
| 1500 | 2989572 | 2966972 | 2960737 | 2934240 | 2920992 |
| 1600 | 2989572 | 2969310 | 2962296 | 2935799 | 2920992 |
| 1700 | 2987234 | 2968530 | 2962296 | 2935799 | 2920213 |
| 1800 | 2988793 | 2967751 | 2962296 | 2936578 | 2917875 |
| 1900 | 2988013 | 2965413 | 2962296 | 2936578 | 2917095 |
| 2000 | 2986455 | 2966192 | 2960737 | 2938137 | 2916316 |

Table 2 presents a detailed CP cost analysis of the STFOA-CPP technique based on the Internet2 topology with 10 controllers. The results show that the STFOA-CPP method achieved effective outcomes with minimal cost compared against other methods under all fitness evaluations. For example, with 100 fitness evaluations, the STFOA-CPP approach demonstrated a lower cost of 2662686, whereas the BBO, BAT, FWA, and FE-QPSO approaches showed higher costs of 2760489, 2716320, 2705278, and 2697390, respectively. With 200 fitness evaluations, the STFOA-CPP method had a lower cost of 2627982, whereas the BBO, BAT, FWA, and FE-QPSO techniques showed higher costs of 2755756, 2713165, 2705278, and 2675306, respectively. Finally, with 300 fitness evaluations, the STFOA-CPP approach showed a lower cost of 2598010 compared to the BBO, BAT, FWA, and FE-QPSO techniques, which demonstrated higher costs of 2751024, 2702123, 2698968, and 2664264, respectively.

Table 3 provides a CP cost analysis of the STFOA-CPP method compared to existing approaches based on the Internet2 topology with 15 controllers. The results show that the STFOA-CPP approach

achieved effective outcomes with minimal cost compared to other methods under all fitness evaluations. For example, with 100 fitness evaluations, the STFOA-CPP approach demonstrated a lower cost of 2670357, whereas the BBO, BAT, FWA, and FE-QPSO models had higher costs of 2602981, 2666334, 2722648, and 2657284, respectively. With 200 fitness evaluations, the STFOA-CPP method had a lower cost of 2684435, whereas the BBO, BAT, FWA, and FE-QPSO models showed higher costs of 2610021, 2670357, 2721643, and 2670357, respectively. Finally, with 300 fitness evaluations, the STFOA-CPP model had a lower cost of 2703542 compared to the BBO, BAT, FWA, and FE-QPSO methods, with higher costs of 2615049, 2672368, 2721643, and 2680413, respectively.

**Table 2:** CP cost analysis of STFOA-CPP approach based on Internet2 topology with 10 controllers

| Cost: Internet2 topology, controllers = 10 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 0 | 2755756 | 2752601 | 2698968 | 2755756 | 2683193 |
| 100 | 2760489 | 2716320 | 2705278 | 2697390 | 2662686 |
| 200 | 2755756 | 2713165 | 2705278 | 2675306 | 2627982 |
| 300 | 2751024 | 2702123 | 2698968 | 2664264 | 2598010 |
| 400 | 2743137 | 2698968 | 2694235 | 2650066 | 2577503 |
| 500 | 2743137 | 2692658 | 2689503 | 2637447 | 2566461 |
| 600 | 2739982 | 2698968 | 2684771 | 2623250 | 2558574 |
| 700 | 2733672 | 2695813 | 2683193 | 2613785 | 2555419 |
| 800 | 2732094 | 2700545 | 2684771 | 2599588 | 2555419 |
| 900 | 2732094 | 2700545 | 2680038 | 2585391 | 2547532 |
| 1000 | 2725785 | 2700545 | 2680038 | 2585391 | 2547532 |
| 1100 | 2727362 | 2697390 | 2678461 | 2582236 | 2547532 |
| 1200 | 2727362 | 2697390 | 2678461 | 2575926 | 2542799 |
| 1300 | 2727362 | 2697390 | 2667419 | 2558574 | 2542799 |
| 1400 | 2725785 | 2697390 | 2667419 | 2556996 | 2542799 |
| 1500 | 2716320 | 2695813 | 2667419 | 2556996 | 2534912 |
| 1600 | 2711587 | 2698968 | 2665841 | 2555419 | 2531757 |
| 1700 | 2710010 | 2697390 | 2664264 | 2555419 | 2528602 |
| 1800 | 2705278 | 2698968 | 2661109 | 2552264 | 2528602 |
| 1900 | 2706855 | 2694235 | 2657954 | 2552264 | 2527025 |
| 2000 | 2708433 | 2692658 | 2659531 | 2552264 | 2523870 |

**Table 3:** CP cost analysis of STFOA-CPP approach based on Internet2 topology with 15 controllers

| Cost: Internet2 topology, controllers = 15 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 0 | 2599965 | 2650245 | 2720637 | 2600970 | 2601976 |

(Continued)

**Table 3:** Continued

| Cost: Internet2 topology, controllers = 15 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 100 | 2602981 | 2666334 | 2722648 | 2657284 | 2670357 |
| 200 | 2610021 | 2670357 | 2721643 | 2670357 | 2684435 |
| 300 | 2615049 | 2672368 | 2721643 | 2680413 | 2703542 |
| 400 | 2619071 | 2677396 | 2723654 | 2694491 | 2716615 |
| 500 | 2618066 | 2676390 | 2724659 | 2704547 | 2735721 |
| 600 | 2618066 | 2677396 | 2723654 | 2721643 | 2747788 |
| 700 | 2628122 | 2687452 | 2723654 | 2720637 | 2755833 |
| 800 | 2634155 | 2689463 | 2723654 | 2728682 | 2764883 |
| 900 | 2630133 | 2689463 | 2723654 | 2735721 | 2767900 |
| 1000 | 2637172 | 2689463 | 2723654 | 2739743 | 2774939 |
| 1100 | 2637172 | 2687452 | 2723654 | 2747788 | 2778962 |
| 1200 | 2638178 | 2688458 | 2723654 | 2752816 | 2781979 |
| 1300 | 2645217 | 2691474 | 2721643 | 2756839 | 2781979 |
| 1400 | 2644211 | 2693486 | 2722648 | 2761867 | 2786001 |
| 1500 | 2646222 | 2693486 | 2723654 | 2761867 | 2787007 |
| 1600 | 2647228 | 2693486 | 2723654 | 2767900 | 2790023 |
| 1700 | 2647228 | 2694491 | 2723654 | 2765889 | 2790023 |
| 1800 | 2647228 | 2696502 | 2723654 | 2767900 | 2791029 |
| 1900 | 2647228 | 2699519 | 2723654 | 2767900 | 2793040 |
| 2000 | 2651250 | 2702536 | 2726671 | 2766895 | 2795051 |

Table 4 shows a detailed CP cost study of the STFOA-CPP approach based on the PlanetV2 topology with five controllers. The results show that the STFOA-CPP technique attained effective outcomes with minimal cost compared to other models under all fitness evaluations. For instance, with 100 fitness evaluations, the STFOA-CPP method attained the lower cost of 0.1388, whereas the BBO, BAT, FWA, and FE-QPSO models showed higher costs of 0.1540, 0.1545, 0.1414, and 0.1450, respectively. With 200 fitness evaluations, the STFOA-CPP approach attained the lower cost of 0.1378, whereas the BBO, BAT, FWA, and FE-QPSO models attained higher costs of 0.1528, 0.1543, 0.1433, and 0.1424, respectively. Finally, with 300 fitness evaluations, the STFOA-CPP model showed a lower cost of 0.1369 compared to the BBO, BAT, FWA, and FE-QPSO models, which attained higher costs of 0.1519, 0.1540, 0.1424, and 0.1390, respectively.

**Table 4:** CP cost analysis of STFOA-CPP approach based on PlanetV2 topology with five controllers

| Cost: PlanetV2 topology, controllers = 5 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 0 | 0.1562 | 0.1545 | 0.1478 | 0.1569 | 0.1412 |
| 100 | 0.1540 | 0.1545 | 0.1414 | 0.1450 | 0.1388 |
| 200 | 0.1528 | 0.1543 | 0.1433 | 0.1424 | 0.1378 |

(Continued)

**Table 4:** Continued

| Cost: PlanetV2 topology, controllers = 5 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 300 | 0.1519 | 0.1540 | 0.1424 | 0.1390 | 0.1369 |
| 400 | 0.1517 | 0.1543 | 0.1424 | 0.1383 | 0.1359 |
| 500 | 0.1514 | 0.1543 | 0.1419 | 0.1381 | 0.1340 |
| 600 | 0.1502 | 0.1540 | 0.1419 | 0.1374 | 0.1338 |
| 700 | 0.1495 | 0.1540 | 0.1414 | 0.1376 | 0.1335 |
| 800 | 0.1495 | 0.1540 | 0.1414 | 0.1369 | 0.1333 |
| 900 | 0.1481 | 0.1543 | 0.1412 | 0.1369 | 0.1333 |
| 1000 | 0.1476 | 0.1543 | 0.1412 | 0.1369 | 0.1333 |
| 1100 | 0.1474 | 0.1543 | 0.1405 | 0.1364 | 0.1333 |
| 1200 | 0.1474 | 0.1543 | 0.1405 | 0.1364 | 0.1333 |
| 1300 | 0.1474 | 0.1540 | 0.1405 | 0.1357 | 0.1331 |
| 1400 | 0.1471 | 0.1540 | 0.1405 | 0.1355 | 0.1328 |
| 1500 | 0.1471 | 0.1538 | 0.1405 | 0.1355 | 0.1328 |
| 1600 | 0.1471 | 0.1538 | 0.1395 | 0.1357 | 0.1328 |
| 1700 | 0.1466 | 0.1538 | 0.1393 | 0.1355 | 0.1328 |
| 1800 | 0.1459 | 0.1536 | 0.1388 | 0.1359 | 0.1324 |
| 1900 | 0.1459 | 0.1536 | 0.1388 | 0.1357 | 0.1326 |
| 2000 | 0.1447 | 0.1536 | 0.1390 | 0.1352 | 0.1326 |
| 2100 | 0.1447 | 0.1536 | 0.1395 | 0.1357 | 0.1326 |
| 2200 | 0.1445 | 0.1531 | 0.1393 | 0.1357 | 0.1326 |
| 2300 | 0.1445 | 0.1536 | 0.1390 | 0.1357 | 0.1321 |
| 2400 | 0.1447 | 0.1536 | 0.1390 | 0.1355 | 0.1319 |

Table 5 presents a CP cost inspection of the STFOA-CPP algorithm based on the PlanetV2 topology with 10 controllers. The results show that the STFOA-CPP method attained effective outcomes with minimal cost compared to other models under all fitness evaluations.

**Table 5:** CP cost analysis of STFOA-CPP approach based on PlanetV2 topology with 10 controllers

| Cost: PlanetV2 topology, controllers = 10 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 0 | 0.1749 | 0.1689 | 0.1696 | 0.1744 | 0.1674 |
| 100 | 0.1746 | 0.1691 | 0.1685 | 0.1696 | 0.1654 |
| 200 | 0.1741 | 0.1691 | 0.1680 | 0.1673 | 0.1631 |
| 300 | 0.1732 | 0.1690 | 0.1680 | 0.1649 | 0.1619 |
| 400 | 0.1732 | 0.1690 | 0.1673 | 0.1636 | 0.1611 |
| 500 | 0.1727 | 0.1688 | 0.1672 | 0.1620 | 0.1599 |
| 600 | 0.1721 | 0.1687 | 0.1666 | 0.1616 | 0.1594 |
| 700 | 0.1720 | 0.1687 | 0.1666 | 0.1611 | 0.1588 |

(Continued)

**Table 5:** Continued

| Cost: PlanetV2 topology, controllers = 10 | | | | | |
|---|---|---|---|---|---|
| Function evaluations | BBO | BAT | FWA | FE-QPSO | STFOA-CPP |
| 800 | 0.1718 | 0.1687 | 0.1665 | 0.1599 | 0.1582 |
| 900 | 0.1708 | 0.1687 | 0.1665 | 0.1601 | 0.1582 |
| 1000 | 0.1711 | 0.1687 | 0.1662 | 0.1596 | 0.1574 |
| 1100 | 0.1702 | 0.1687 | 0.1662 | 0.1590 | 0.1570 |
| 1200 | 0.1702 | 0.1688 | 0.1660 | 0.1589 | 0.1567 |
| 1300 | 0.1702 | 0.1688 | 0.1660 | 0.1589 | 0.1563 |
| 1400 | 0.1702 | 0.1689 | 0.1660 | 0.1583 | 0.1564 |
| 1500 | 0.1705 | 0.1690 | 0.1660 | 0.1586 | 0.1557 |
| 1600 | 0.1698 | 0.1689 | 0.1660 | 0.1583 | 0.1558 |
| 1700 | 0.1690 | 0.1689 | 0.1659 | 0.1583 | 0.1558 |
| 1800 | 0.1686 | 0.1689 | 0.1658 | 0.1582 | 0.1557 |
| 1900 | 0.1686 | 0.1689 | 0.1657 | 0.1576 | 0.1557 |
| 2000 | 0.1688 | 0.1689 | 0.1657 | 0.1576 | 0.1557 |
| 2100 | 0.1683 | 0.1689 | 0.1657 | 0.1572 | 0.1557 |
| 2200 | 0.1683 | 0.1689 | 0.1657 | 0.1572 | 0.1556 |
| 2300 | 0.1683 | 0.1689 | 0.1657 | 0.1576 | 0.1556 |
| 2400 | 0.1686 | 0.1689 | 0.1657 | 0.1576 | 0.1556 |

Fig. 2 shows the CP cost review of the STFOA-CPP method based on the PlanetV2 topology with three different quantities of controllers. The results showed that the STFOA-CPP method attained effective outcomes with minimal cost compared to other models under all fitness evaluations. For instance, with 100 fitness evaluations, the STFOA-CPP algorithm showed a lower cost of 0.1552, whereas the BBO, BAT, FWA, and FE-QPSO models have attained higher costs of 0.1597, 0.1564, 0.1565, and 0.1583, respectively. With 200 fitness evaluations, the STFOA-CPP technique attained the lower cost of 0.1543, whereas the BBO, BAT, FWA, and FE-QPSO methods showed higher costs of 0.1596, 0.1561, 0.1564, and 0.1565, respectively. Finally, with 300 fitness evaluations, the STFOA-CPP algorithm demonstrated a lower cost of 0.1537 compared to the BBO, BAT, FWA, and FE-QPSO models, which attained higher costs of 0.1592, 0.1562, 0.1564, and 0.1522, respectively.

A detailed cost inspection of the STFOA-CPP model compared against recent models based on the Internet2 topology with varying numbers of controllers is given in Fig. 3. The results show that the STFOA-CPP model achieved effective outcomes with lower cost values for all quantities of controllers. For instance, with four controllers, the STFOA-CPP model exhibited a decreased cost of 2736740, whereas the BBO, BAT, FWA, and FE-QPSO models showed higher costs of 3144657, 3000686, 2888709, and 2808726, respectively. With five controllers, the STFOA-CPP approach demonstrated a decreased cost of 2440801, whereas the BBO, BAT, FWA, and FE-QPSO techniques showed higher costs of 3040678, 2952696, 2880711, and 2744739, respectively. Finally, with six controllers, the STFOA-CPP method displayed a decreased cost of 2352819 compared to the BBO, BAT, FWA, and FE-QPSO methods, which demonstrated higher costs of 2896708, 2760736, 2696749, and 2608767, respectively.

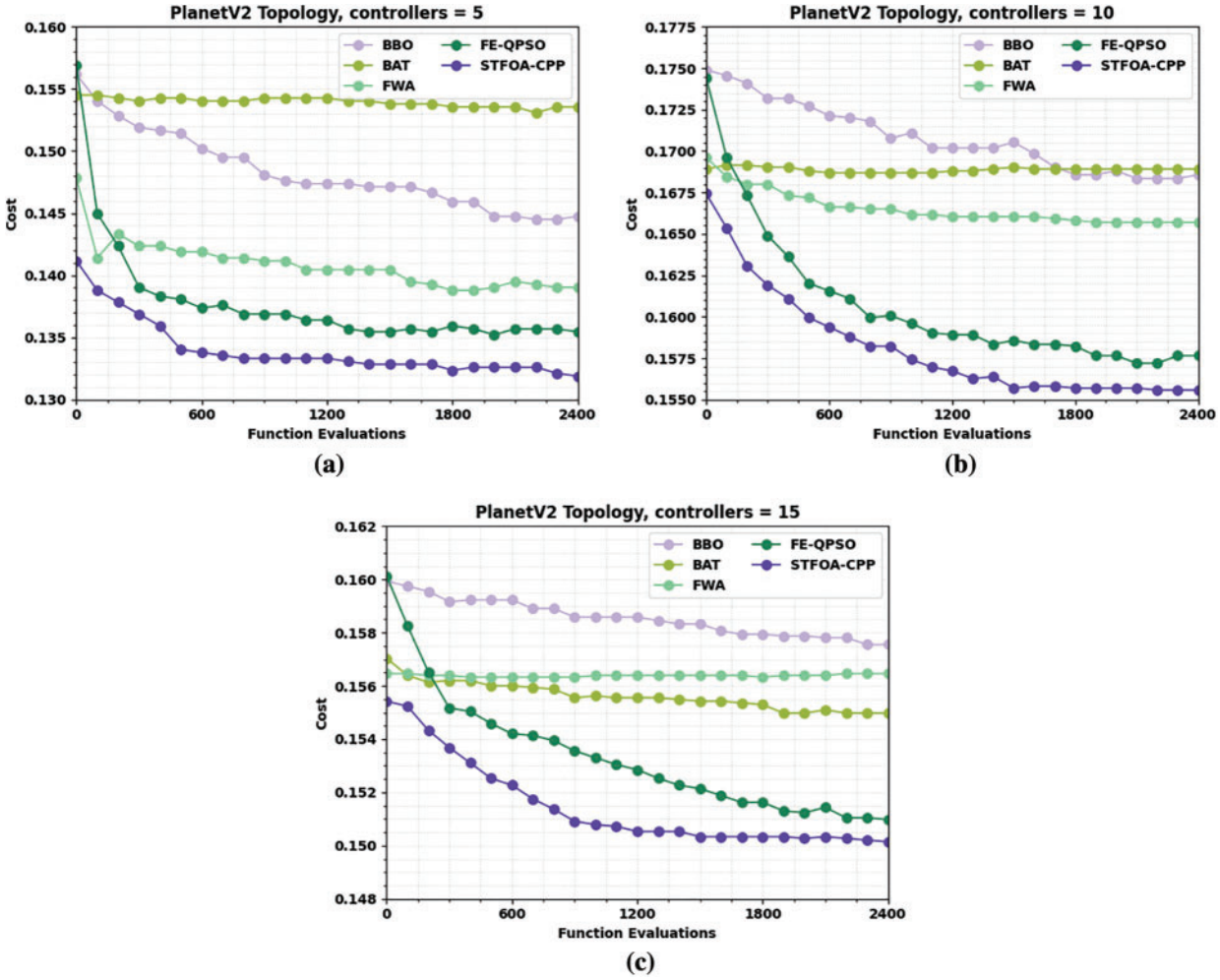


**Figure 2:** CP Cost analysis of STFOA-CPP approach based on PlanetV2 topology (a) controllers = 5, (b) controllers = 10, and (c) controllers = 15

A detailed cost inspection of the STFOA-CPP model compared against recent models based on the PlanetV2 topology with varying numbers of controllers is given in Fig. 4. The results show that the STFOA-CPP approach has effective outcomes with lower cost values under all controllers. For example, with for controllers, the STFOA-CPP model exhibited a decreased cost of 0.1354, whereas the BBO, BAT, FWA, and FE-QPSO techniques showed higher costs of 0.1541, 0.1514, 0.1516, and 0.1483, respectively. With five controllers, the STFOA-CPP method exhibited a decreased cost of 0.1346, whereas the BBO, BAT, FWA, and FE-QPSO models attained higher costs of 0.1450, 0.1541, 0.1390, and 0.1359, respectively. Finally, with six controllers, the STFOA-CPP model exhibited a decreased cost of 0.1401 compared to the BBO, BAT, FWA, and FE-QPSO models, which demonstrated higher costs of 0.1461, 0.1808, 0.1415, and 0.1361, respectively.

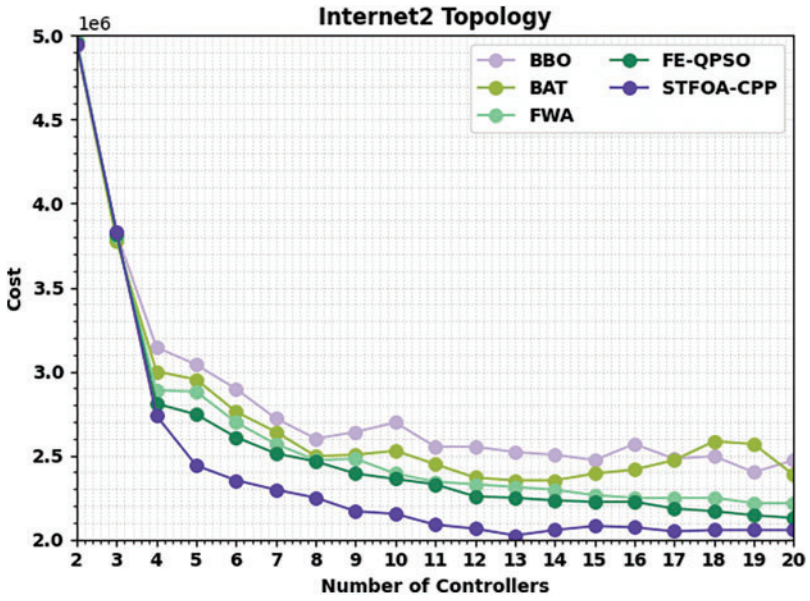


**Figure 3:** Cost analysis of STFOA-CPP approach based on Internet2 topology

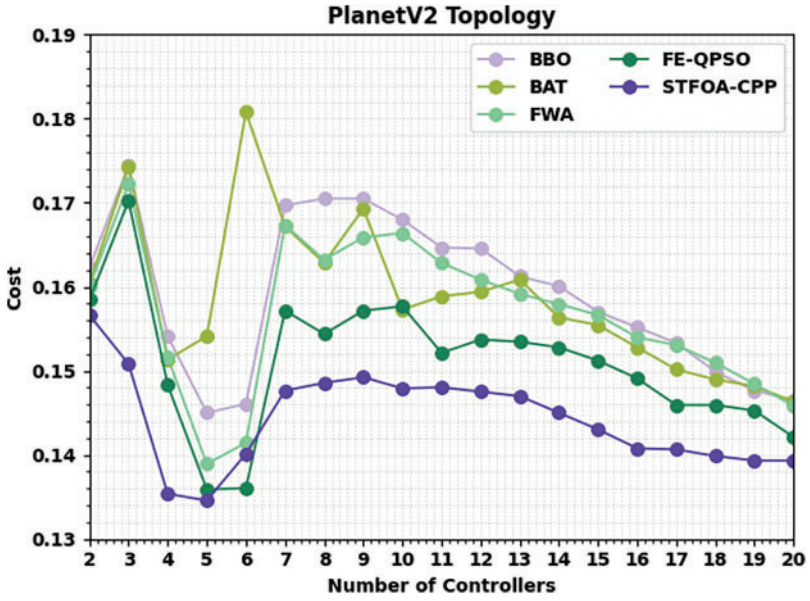


**Figure 4:** Cost analysis of STFOA-CPP approach based on PlanetV2 topology

A detailed comparison study of the STFOA-CPP technique against existing models is given in Fig. 5. The results show that the STFOA-CPP technique demonstrates enhanced performance over the baseline and existing techniques. For example, at 50 days, the presented STFOA-CPP model attained a higher trust value of 0.9473, whereas the baseline and existing methods attained lower trust values of 0.9382 and 0.9427, respectively. Therefore, the proposed STFOA-CPP model can be employed for optimal controller placement in SDN.

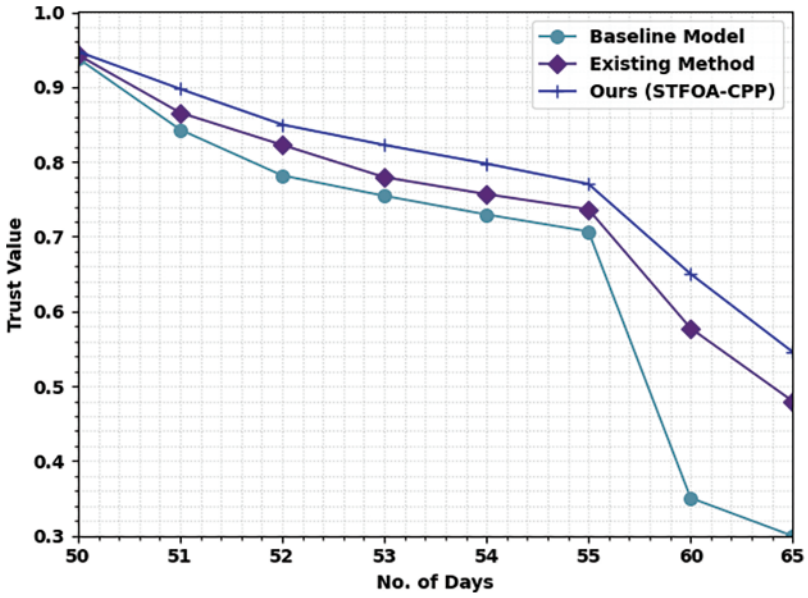


**Figure 5:** Comparative trust value analysis of the STFOA-CPP approach

## 5 Conclusion

In this study, a new STFOA-CCP technique has been developed for CPP in the SDN environment. The goal of the STFOA-CPP technique is to maximize lifetime, network connectivity, and load balancing with minimal latency. The STFOA-CPP technique is based on the sea turtles' food-searching characteristics of tracking the odour path of DMS released by their food sources. Furthermore, the presented STFOA-CPP technique can adapt to the number of controllers necessary and the switch to controller mapping to adjustable network traffic. Finally, the blockchain can inspect data integrity, determine significantly malicious input, and improve the robust nature of developing a trust relationship between several nodes in the SDN. To demonstrate the enhanced performance of the STFOA-CPP technique, a wide-ranging experimental analysis was carried out. The extensive comparison study highlighted the improved outcomes of the STFOA-CPP technique over other recent approaches. Therefore, the STFOA-CPP technique can be applied to solve CPP in the SDN environment.

**Funding Statement:** The author received no specific funding for this study.

**Conflicts of Interest:** The author declares that they have no conflicts of interest to report regarding the present study.

## References

[1] A. Kumari and A. S. Sairam, "Controller placement problem in software-defined networking: A survey," *Networks*, vol. 78, no. 2, pp. 195–223, 2021.

[2] M. Dhar, A. Debnath, B. K. Bhattacharyya, M. K. Debbarma and S. Debbarma, "A comprehensive study of different objectives and solutions of controller placement problem in software-defined networks," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 5, pp. e4440, 2022.

[3] A. Shirmarz and A. Ghaffari, "Taxonomy of controller placement problem (CPP) optimization in software defined network (SDN): A survey," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 12, pp. 10473–10498, 2021.
[4] G. Schütz and J. A. Martins, "A comprehensive approach for optimizing controller placement in software-defined networks," *Computer Communications*, vol. 159, no. 8, pp. 198–205, 2020.
[5] J. Chen, Y. -J. Xiong, X. Qiu, D. He, H. Yin *et al.,* "A cross entropy based approach to minimum propagation latency for controller placement in software defined network," *Computer Communications*, vol. 191, no. 99, pp. 133–144, 2022.
[6] W. Li, J. Tan and Y. Wang, "A framework of blockchain-based collaborative intrusion detection in software defined networking," in *Network and System Security, NSS 2020, LNCS 12570*, Cham: Springer, pp. 261–276, 2020.
[7] M. Eskandari, Z. H. Janjua, M. Vecchio and F. Antonelli, "Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices," *IEEE Internet of Things Journal*, vol. 7, no. 8, pp. 6882–6897, 2020.
[8] W. Li, S. Tug, W. Meng and Y. Wang, "Designing collaborative blockchained signature-based intrusion detection in IoT environments," *Future Generation Computer Systems*, vol. 96, no. 3, pp. 481–489, 2019.
[9] A. A. Seyedkolaei, S. A. H. Seno, A. Moradi and R. Budiarto, "Cost-effective survivable controller placement in software-defined networks," *IEEE Access*, vol. 9, pp. 129130–129140, 2021.
[10] I. Koutsopoulos, "Learning the optimal controller placement in mobile software-defined networks," in *2022 IEEE 23rd Int. Symp. on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Belfast, United Kingdom, pp. 70–79, 2022.
[11] N. Samarji and M. Salamah, "A fault tolerance metaheuristic-based scheme for controller placement problem in wireless software-defined networks," *International Journal of Communication Systems*, vol. 34, no. 4, pp. e4624, 2021.
[12] Y. Wu, S. Zhou, Y. Wei and S. Leng, "Deep reinforcement learning for controller placement in software defined network," in *IEEE INFOCOM, 2020—IEEE Conf. on Computer Communications Workshops (INFOCOM WKSHPS)*, Toronto, ON, Canada, pp. 1254–1259, 2020.
[13] T. Hu, Q. Ren, P. Yi, Z. Li, J. Lan *et al.,* "An efficient approach to robust controller placement for link failures in software-defined networks," *Future Generation Computer Systems*, vol. 124, no. 1, pp. 187–205, 2021.
[14] Y. Li, S. Guan, C. Zhang and W. Sun, "Parameter optimization model of heuristic algorithms for controller placement problem in large-scale SDN," *IEEE Access*, vol. 8, pp. 151668–151680, 2020.
[15] N. firouz, M. Masdari, A. B. Sangar and K. Majidzadeh, "A hybrid multi-objective algorithm for imbalanced controller placement in software-defined networks," *Journal of Network and Systems Management*, vol. 30, no. 3, pp. 51, 2022.
[16] A. Jalili, M. Keshtgari and R. Akbari, "A new framework for reliable control placement in software-defined networks based on multi-criteria clustering approach," *Soft Computing*, vol. 24, no. 4, pp. 2897–2916, 2020.
[17] S. Kotachi, T. Sato, R. Shinkuma and E. Oki, "Fault-tolerant controller placement model by distributing switch load among multiple controllers in software-defined network," *IEICE Transactions on Communications*, vol. 105, no. 5, pp. 533–544, 2022.
[18] S. Guan, J. Li, Y. Li and Z. Wang, "A multi-controller placement method for software defined network based on improved firefly algorithm," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 7, pp. e4482, 2022.
[19] C. S. Endres and K. J. Lohmann, "Perception of dimethyl sulfide (DMS) by loggerhead sea turtles: A possible mechanism for locating high-productivity oceanic regions for foraging," *Journal of Experimental Biology*, vol. 215, no. 20, pp. 3535–3538, 2012.
[20] D. Tansui and A. Thammano, "Hybrid nature-inspired optimization algorithm: Hydrozoan and sea turtle foraging algorithms for solving continuous optimization problems," *IEEE Access*, vol. 8, pp. 65780–65800, 2020.

[21] W. Li, Y. Wang, W. Meng, J. Li and C. Su, "BlockCSDN: Towards blockchain-based collaborative intrusion detection in software defined networking," *IEICE Transactions on Information and Systems*, vol. 105, no. 2, pp. 272–279, 2022.
[22] D. Simon, "Biogeography-based optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 6, pp. 702–713, 2008.
[23] X. -S. Yang, "A new metaheuristic bat-inspired algorithm," In: J. R. González, D. A. Pelta, C. Cruz, G. Terrazas, N. Krasnogor (Eds.), *Nature Inspired Cooperative Strategies for Optimization (NICSO 2010)*, Berlin, Heidelberg: Springer, Berlin, Heidelberg, pp. 65–74, 2010.
[24] Y. Tan and Y. Zhu, "Fireworks algorithm for optimization," *Advances in Swarm Intelligence*, vol. 6145, pp. 355–364, 2010.
[25] Q. Zhang, L. Haolun, L. Yanli, S. Ouyang, C. Fang *et al.,* "A new quantum particle swarm optimization algorithm for controller placement problem in software-defined networking," *Computers and Electrical Engineering*, vol. 95, no. 1, pp. 107456, 2021.