Received 25 February 2022; revised 17 April 2022; accepted 1 May 2022. Date of publication 6 May 2022; date of current version 17 May 2022.

Digital Object Identifier 10.1109/OJCOMS.2022.3172551

An Adapted Nondominated Sorting Genetic Algorithm III (NSGA-III) With Repair-Based Operator for Solving Controller Placement Problem in Software-Defined Wide Area Networks

OLADIPUPO ADEKOYA (Graduate Student Member, IEEE), AND ADEL ANEIBA

Computing Engineering and Built Environment Department, Birmingham City University, Birmingham B5 5JU, U.K.

CORRESPONDING AUTHOR: O. ADEKOYA (e-mail: oladipupo.adekoya@mail.bcu.ac.uk)

ABSTRACT Optimum controller placement in the presence of several conflicting objectives has received significant attention in the Software-Defined Wide Area Network (SD-WAN) deployment. Multi-objective evolutionary algorithms, like Non-dominated Sorting Genetic Algorithm II (NSGA-II) and Multi-objective Particle Swamp Optimization (MOPSO), have proved helpful in solving Controller Placement Problem (CPP) in SD-WAN. However, these algorithms were associated with the challenge of scalability (when there are more than three objectives) for optimization in the SD-WAN. Hence, this study proposed an adapted NSGA-III (A-NSGA-III) to resolve the scalability challenges associated with NSGA-II and MOPSO algorithms in the presence of more than three objectives. This study developed and introduced a repair-based operator into the existing Mechanical Engineering based NSGA-III to propose the A-NSGA-III for optimal controller placement in the SD-WAN. The proposed A-NSGA-III, the NSGA-II and MOPSO algorithms were subjected to evaluation using datasets from Internet2 OS3E WAN topology with six objective functions. The Hypervolume indicator, Percentage Coefficient of Variation (PCV), the percentage difference and the Parallel Coordinate Plots (PCP) confirmed that the proposed A-NSGA-III exhibited high convergence and diversification than the NSGA-II and MOPSO algorithms in the presence of scalability challenge (when the number of objective function exceeded three). The result confirmed that the proposed A-NSGA-III solved the scalability challenges associated with the optimal Controller Placement in the SD-WAN. Hence, A-NSGA-III was recommended over NSGA-II and MOPSO algorithms, subject to the confirmation usage conditions.

INDEX TERMS Controller placement, adapted NSGA-III, repair operator-based mechanism, Pareto-frontier, SD-WAN.

I. INTRODUCTION

SOFTWARE-DEFINED Wide Area Network (SD-WAN) architecture in recent years has become emerging technology that provides various benefits to the industry and academia. In contrast to conventional network architectures, which combine the data plane and the control plane on the same plane, the SD-WAN architecture separates the control plane, and the data plane [1]. The SD-WAN enables centralised network management, which significantly improves the way the network is managed [2]. Although, the SD-WAN architecture originally comes with the controller being logically centralised within the topology. However, several related literature studies suggested that the controller should be logically centralised and physically distributed to enhance various performance metrics. Such metrics include deployment cost, load balancing, resilience, scalability, switch-to-controller latency, and inter-controller latency, among others [3].

The initial centralised controller (also known as Software Defined Network) approach is less efficient in a large-scale

network like SD-WAN [4]. When this controller is deployed in such less efficient environment, the centralised controller would be associated with scalability challenge (optimizing more than three competing objectives simultaneously) and low latency performance. However, the work of [5] had suggested the deployment of multiple controllers in SD-WAN for the correction of the aforementioned challenges. The identification of the appropriate location for the controller placement within the network topology among several connected nodes to meet the network requirements has created a controller placement challenge which is known as Non-deterministic Polynomial hardness (NP-hard) [6].

Given the several objectives, the simultaneous optimization of these objectives (called multi/many-objectives optimization or scalability) has been considered a challenge in SD-WAN. The Pareto-Optimal solutions [7] had been implemented to place a trade-off for optimal solutions within the SD-WAN. The Controller Placement Problem (CPP), also known as NP-hard [8], considers n switches (nodes) to position k number of controllers, such that $(k < n)$, to create the most optimum network in the presence of several conflicting objectives [5].

Mathematical optimization techniques (such as mixed-integer linear programming and dynamic programming) are optimization techniques for solving controller placement in SD-WAN. However, these techniques could quickly get stuck in local optimal due to its lack of memory during the algorithm operations. It may take tens of minutes or more which is not an effective solution to manage a large-scale environment with dynamically changing network conditions [7]. The evolutionary algorithm, for instance, Non-dominated Sorting Genetic Algorithm II (NSGA-II) [9] and Swarm Intelligence algorithms, such as Multi-Objective Particle Swamp Optimization (MOPSO) [10] are examples of those algorithms developed in the literature to optimize CPP. The above two evolutionary techniques are problem-dependent and utilise non-dominating sorting and crowding distance techniques to ensure convergence to the true Pareto Front and preserve diversity among the solutions. However, these approaches are characterised by diversification challenges, significantly when the objectives exceed three [9] and the population is over a hundred ([11] and [12]). Similarly, all solutions in these approaches would become non-dominated as most or almost all solutions would domicile in the first front. Consequent to the aforementioned drawbacks, an algorithm that would ensure diverse solutions among the set of solutions, should be considered [13]. Hence, there is a need to explore an alternative and improved method for a vast network environment where the condition of the network (traffic pattern and bandwidth usage) changes dynamically.

This study has adapted an evolutionary algorithm called Non-dominated Sorting Genetic Algorithm III (NSGA-III). The NSGA-III algorithm was adopted from the Mechanical Engineering discipline. In order to accomplish the aim of this study, the NSGA-III was modified by introducing a repair-based operator mechanism to effectively solve the identified challenge of controller placement in an SD-WAN environment.

The main contributions of this research are summarised as follows.

- Limited to the literature, this research proposes the maiden adaption of NSGA-III to solve the problem of controller placement in a SD-WAN environment.
- The study shall develop a repair operator-based mechanism to replace the continuous optimization characteristics with discrete optimization characteristics in the existing NSGA-III.
- The study incorporates the developed repair operator-based mechanism onto the adapted NSGA-III for the optimal placement of controllers in SD-WAN, which ensures both convergence and well-diversified solutions among the non-dominated solutions.
- The study adapts a many-objective evolutionary algorithm that shows the trade-offs among the objective functions.

The contents of this manuscript are organized as follows. Section II surveys the related works. Section III explains the scenario of many-objective controller placement. Section IV proposes an efficient adapted NSGA-III that addresses the controller placement problem in SD-WAN. Finally, the experimentation with analytical results and the conclusion of this research are described in Sections V, VI, and VII, respectively.

II. RELATED WORKS

The CPP in SD-WAN had been well explored in the literature. However, there are still many ongoing types of research in this regard, especially when there is a need to simultaneously optimize several conflicting objectives to meet the SD-WAN requirements. Like the Facility Location Problem (FLP), the CPP is a Non-Deterministic Polynomial (NP-hard) problem. Random placement of controllers in an accessible location would give rise to the network overhead and reduce the performance of SD-WAN based infrastructure [14].

The work of [15] considered SDN controller security, although this is out of the scope of this work. However, the security knowledge of controllers in SDN will enhance the overall performance of controllers. The work of [16] explored the reaction of the controller placement on average and the worst-case switch-to-controller latencies in small-to-medium size networks and static environments. The study formulated CPP as a minimum k-center problem and minimum k-median problem to minimise the average and worst-case node-to-controller latency. The number of controllers is represented as k. Similarly, the k-center problem is a facility location problem similar to the optimization problem in operation research [14]. The exhaustive approach considered in the study evaluated all possible combinations of controller placement concerning node-to-controller latency for the average and worst-case scenarios. Although, this approach guarantees optimal solutions concerning latency. However, a practical, computational and quick approach is

required to manage large-scale and dynamic environments such as SD-WAN.

The study of [17] improved on the developed solution of [16] by presenting a Capacitated Controller Placement Problem (CCPP). The study considered the load of the control plane as a second objective in addition to the propagation latency that was considered by [16]. The authors established that load is also a key component metric that needs proper consideration as much as latency when finding the optimal location of controllers. It was further shown that the propagation latency, as a metric, is not enough to evaluate controller placement. Meanwhile, the study assumed that limited data plane devices could only be assigned to a controller since the capacity of each controller in the network is limited by server capacity and bandwidths of the links. This study ignored performance metrics such as cost, fault-tolerance, and inter-controller latency, which are all critical in real networks. Moreover, the dynamic programming proposed in the approach is computationally expensive and only suitable in small-medium-sized networks or at the design stage where time is not a limiting criterion.

The study of [18] explored the optimal placement of controllers in SD-WAN to enhance failure tolerance and resiliency. The work simultaneously optimized several performance metrics and concluded that a trade-off solution existed between these metrics. It was concluded that a single controller placement that satisfied several objectives, especially when these objectives are conflicting in nature, does not exist. An exhaustive approach, although based on Pareto-optimal controller placement, was proposed in the [18] to evaluate possible controller placement. Although, the process was confirmed to provide a true Pareto optimal solution, it lacked a heuristics solution which was a severe drawback to the approach. Such defects include the inability to provide well-diversified Pareto Front solutions when the Pareto-optimal is non-convex. Similarly, the technique was not efficient in an SD-WAN environment where network conditions (like bandwidth usage and traffic pattern) change dynamically.

The study of [19] investigated SD-WAN in an environment with dynamic network conditions. The study developed a dynamic controller provisioning problem that is contrary to the static environment examined by [16]. The location of controllers in this approach varies over time due to the network's traffic flows. The study presented a strategy that dynamically conforms to the number of controllers during network operations. Consequently, each controller was associated with a set of switches based on dynamic network changes to ensure that communication budget and minimal flow setup time were guaranteed. However, this developed greedy knapsack and simulated annealing algorithm may get stuck at the local optimal. Consequently, this may not produce a well-diversified solution set when the number of conflicting objectives is more than three.

Unlike the mathematical approach, the evolutionary algorithm can address multi/many-objective optimization problems. This approach is capable of providing multiple solutions in one simulation run [9]. The literature has established the development of several multi-objective evolutionary algorithms to address the CPP in SD-WAN. The work of [10] defined a global latency CPP where latency between controllers and load on the controller is being considered as the performance metrics. The study also developed the multi-objective particle swarm optimization algorithm. However, the approach may only be efficient when the scalability is not more than three objectives. Consequently to the violation of this condition, this may lead to a less widely distributed solution across the Pareto Front.

Similarly, the work of [20] developed a controller placement algorithm to handle a three-objective placement strategy. The objectives considered were minimising propagation latency between nodes and the associated controllers, load imbalance, and minimising inter-controllers latency. The NSGA-II was presented to solve the CPP. However, the system may not preserve diverse Pareto Front solutions when the objectives exceed three [9] and the size of the population is more than a hundred [11]. This was due to the crowded distance approach utilised in the algorithm since most solutions would become non-dominated and all domicile on the first front.

The literature had proposed a multi-Objective evolutionary-based algorithm to decide where to place the controller in the company of several or multiple conflicting objectives. Evolutionary algorithms such as NSGA-II and MOPSO are examples of those algorithms developed in the literature to optimize CPP in several conflicting objectives. However, multi-objective evolutionary algorithms that utilise non-dominated sorting and crowding distance for exploitation and exploration are associated with the drawback of not preserving diversity across the Pareto Front solutions and inefficient performance-wise as most solutions domiciles in the first front.

To solve the scalability challenge (the inability to optimise when the number of objective function is more than three) and improve the exploration and exploitation of the candidate solution, this study has presented an A-NSGA-III with a repair operator strategy for solving CPP in SD-WAN. This proposed approach gives improved solutions improvement over the existing NSGA-II and MOPSO frameworks in scalability, convergence, and diversification. The experimentation results conducted, in this study, confirmed the flexibility and efficiency of the proposed algorithm.

III. PROBLEM DEFINITION

The CPP in SD-WAN was investigated in this study. This solution proposed optimal placement of controllers regarding the access network topology such that several network requirements are satisfied simultaneously. While latency between the switch and its connecting controllers forms the most critical CPP situation, other conflicting objectives need to be considered. These objectives include resilience, controller load balancing, and inter-controller latency.

This study proposed a controller placement algorithm as an unconstrained many-objective CPP. The objective functions of this study are introduced in equations (1) through (5). The SD-WAN is constructed as an undirected graph $G = (V, E)$, where V denotes the set of nodes, and E denotes the connections between the nodes. Meanwhile, a distance matrix D with the shortest path latency between each pair of nodes information is also needed to calculate placement. The latency between node i and node j is denoted as d_{ij}. Note that this study had divided the latency in D by the respective graph diameter for the basis of normalization. Given the desired number of controllers, the search space is constrained to a set of $\binom{n}{k}$ placements to get the desired result.

This study refers to placement as k-element, such that k is a subset of V. The search space for the CPP is the k-subset of V, which is a space of all possible solutions. For illustrative instance, given a network topology with 38 nodes and a predefined number of controllers such that $k = 5$, then the set $X = \{4, 14, 17, 28, 33\}$ denotes controller positions ($|P| = 5$). The five controllers, on the assumption, should be placed in nodes 4, 14, 17, 28 and 33. Note that swapping the members' locations in each subset would not give any new combination. Hence, with this instance, the total number of feasible placements in this network is $\binom{38}{5}$. It is assumed that a set of objectives $\{j_1, j_2, \ldots, j_m\}$ is known and is to be minimised. X is Pareto optimal if no alternate placement Y exists in the search place. That is, $\forall_i\, j_i(Y) \leq j_i(X)$ and $j_i(Y) < j_i(X)$ no less than one index i. The reason for addressing the CPP is to locate the Pareto optimal set of the entire search space and the set of objective values of all Pareto optimal placements, which form a set of solutions known as the Pareto Frontier.

A. OBJECTIVE FUNCTIONS

This subsection included an overview of the reviewed objectives. The reader is referred to [20] for additional information on the reviewed objectives. Various conflicting objectives should be considered in evaluating controller placement, given a placement P of controllers. The first two performance metrics give information about the maximum and average switch-to-controller latency. This refers to the connection between the switch and its controller. For each candidate placement $P \in 2^V$ and the predefined distance matrix D, equations (1) and (2) account for the maximum and average switch-to-controller latency, respectively.

$$\pi^{Lat\_max\_S2C}(P) = \max_{v \in V} \min_{p \in P} d_{v,p}, \tag{1}$$

$$\pi^{Lat\_avg\_S2C}(P) = \frac{1}{|V|} \sum_{v \in V} \min_{p \in P} d_{v,p}. \tag{2}$$

In a large size-networks where multiple controllers are deployed, these controllers need to communicate and share information. Consequently, inter-controller latency should be considered to evaluate controller placement, which should also be minimised. Similar to equations (1) and (2), which consider both maximum and average form of latency, equations (3) and (4) do the same but compute the inter-controller latency. This objective has a substantial impact on the coordination of controllers and should be examined in the CPP.

$$\pi^{Lat\_max\_S2C}(P) = \max_{p_1, p_2 \in P} d_{p_1,p_2}, \tag{3}$$

$$\pi^{Lat\_avg\_S2C}(P) = \frac{1}{\binom{|P|}{2}} \sum_{p_1, p_2 \in P} d_{p_1,p_2}. \tag{4}$$

While the latency-based objectives aim for minimum communication paths in the network, controller load balance should also be considered when network operation reliability is sought. Since the rest of the objective functions are minimized, an imbalance metric is presented here in place of a balancing metric for compliance purposes and also to balance the load distribution between controllers [20]. Therefore, for each placement P and controller p, the total number of devices allocated to p when each device connects to its closest controller is referred to as n_p. The imbalance metric denotes the dissimilarity between n_p for two controllers with the lowest and highest allocated nodes and is depicted in equation (5) [20].

$$\pi^{imbalance}(P) = \max_{p \in P} n_p - \min_{p \in P} n_p. \tag{5}$$

This study also considered resilience concerning controller failure as an objective function during the optimal placement of controllers. Assuming that $C = 2^P \backslash \{\emptyset\}$ represents all possible placements left from the outages of up to $(k - 1)$ controllers, then the average switch to controller latency for any failure scenario is represented mathematically in equation (6).

$$\pi^{Lat\_avg\_S2C}(P) = \frac{1}{|C|} \sum_{P \in C} \left(\frac{1}{|V|} \sum_{v \in V} \left(\min_{p \in P} d_{v,p} \right) \right). \tag{6}$$

IV. PROPOSED ADAPTED NSGA-III (A-NSGA-III) FOR SD-WAN CONTROLLER PLACEMENT

The proposed adapted Non-dominated Sorting Genetic Algorithm III (hereafter referred to as A-NSGA-III) with a repair-based operator is presented in this section. The NSGA-III [9] was first developed in the Mechanical Engineering to address more than three-objective optimization problems. The existing NSGA-III could not be applied directly to solve the CPP in SD-WAN because it is limited to continuous optimization problems while the CPP is a discrete optimization problem. Consequently to this limitation, the existing NSGA-III, in application to CPP, did not produce a feasible search space and unique solutions without duplicates. The framework of NSGA-II was used in NSGA-III but with the reference point-based approach and other mechanisms (such as reference points, normalization, association, and niching techniques) to improve convergence and diversity preservation.

The reviewed NSGA-II and MOPSO algorithms that employs Pareto dominance techniques in ranking solutions [9] and crowding distance operators for preserving diversity [21] would not be efficient when they are applied to address scalability challenge (more than three objectives). Even in the early generation, most of the solutions would become non-dominated and lie in the first level layer (first Front), resulting in difficulty to sustain an adequate selection pressure of elite solutions towards optimal solutions. However, [9] and [22] confirmed that the presence of a guidance mechanism such as a clustering operator with a well-distributed reference point in NSGA-III helps to maintain diversity among solutions. Similarly, [23], and [24] confirmed that the self-adaptive update of the reference point set determined from the association state of each reference point over several evaluations makes the algorithm a more computationally fast approach. Consequently, the reference points, association, and niching techniques introduced in the NSGA-III procedure are expected to justify the significant diversification and convergence improvement of A-NSGA-III over NSGA-II.

Literature has established that when the number of objectives exceeds three (scalability challenge) [9], and the size of the population is more than a hundred [11], the NSGA-II and MOPSO do not preserve diversity across the Pareto Optimal set. Consequent to these algorithm's drawbacks, there is evidence to develop an algorithm to select diverse solutions sequel to these characteristics. Hence, this justifies the maiden introduction of the adapted A-NSGA-III in SD-WAN.

A. THE DESCRIPTION OF THE PROPOSED A-NSGA-III

The proposed A-NSGA-III is presented in Algorithm 1 while the discussion follows in this subsection. The A-NSGA-III algorithm is supplied with a structured reference point H and the parent population as P_t. The reference point can either be computed using [25] systematic approach where $H = \binom{M+p-1}{p}$ or be supplied by the client. The H defined the reference points number, the $P_{(t+1)}$ is the next-generation population (output). The reference points are set in advance and created on a unit hyper-plane to ensure they are uniformly distributed across the entire normalized hyper-plane (see Figure 1). Because the reference points obtained above are widely distributed across the normalized hyper-plane, the derived solutions will also be widely distributed towards the Pareto optimum front. This is a requirement that is based on the diversity which is defined concerning the reference points or reference lines [9].

Line 3 of Algorithm 1, X_t (the population set) saves solution in the Front set $\{FR_1, FR_2, \ldots, FR_M\}$ and i is the generation counter which is set to 1. Line 4, the simulated binary crossover and polynomial mutation is applied to the parent population (P_t) to get the offspring population. In line 5, a repair-based operator is applied on the newly formed population set C_t to remove infeasible offspring's solutions. In line 6, the parent population and offspring

Algorithm 1 Proposed Adapted NSGA-III for SD-WAN Controller Placement Problem

1: **Input**: *H uniformly distributed reference points, Y^s or provide aspiration points Y^a parent population P_t*
2: **Output**: P_{t+1}
3: $X_t = \emptyset,\ i = 1$
4: $C_t = Recombination + Mutation$
5: *Apply repair operator mechanism on* C_t
6: $W_t = P_t \cup C_t$
7: $(FR_1, FR_2, ...) = Nondominated - sort(W_t)$
8: *repeat*
9: $X_t = X_t \cup FR_i$ *and* $i = i + 1$
10: *until* $|X_t| \geq N$
11: *Last front to be included*:$FR_l = FR_i$
12: **if** $|S_t| = N$ **then**
13: $P_{t+1} = X_t$, *break*
14: **else**
15: $P_{t+1} = \bigcup_{j=1}^{l-1} F_j$
16: *Points to be chosen from* FR_l:$K = N - |P_{t+1}|$
17: *Perform objectives Normalization and create reference set* Y^s:*Normalize* $(f^n, X_t, Y^r, Y^s, Y^a)$
18: *Associate each member x of* X_t *with a ref point*: $[\pi(s), d(s)]$ = *Associate* $(X_t, Y^r,)$ $\pi(s)$:*closest ref point* d:*distance between s and* $\pi(s)$
19: *Calculate niche count of reference point* $j \in Y^r$: $\rho_j = \sum_{X \in X_t/FR_t} ((\pi(s) = j)\ ?1:0)$
20: *Select K members one at a time from* FR_l *to construct* P_{t+1}:$Niching(K, \rho_j, d, Y^r, F_l, P_{t+1})$
21: *Output Corresponding Placements*
22: **end if**

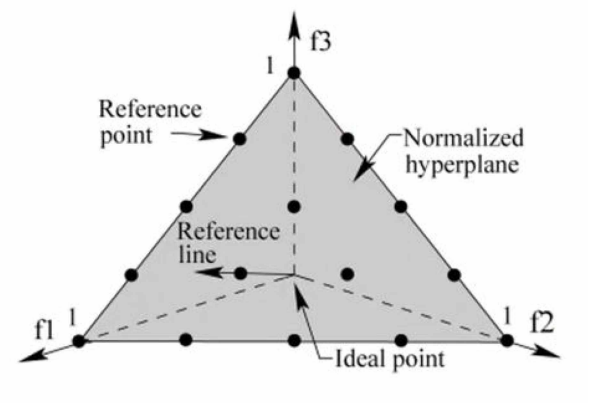

FIGURE 1. Reference point on a Unit hyperplane [9].

are combined and saved in W_t. The non-dominated sorting was carried out on W_t and the solutions were arranged in $\{FR_1, FR_2, \ldots, FR_M\}$ in line 7. The readers are referred to [9] for more information on the NSGA-III and its mechanisms. In line 8, the process of non-dominating sorting is repeated. In contrast, in lines 9 and 10, the solution from the front is included in the population set until the population set is greater than population size N. Line 11 describes the characteristics of the last front to be added before the condition in line 10 could be satisfied. In lines 12 and 13, if the last front is added and the size of X_t = N, then the

algorithm breaks, and the counter increases with 1. In lines 14 and 15, the fronts are added to the next generation $P_{(t+1)}$ except the last front. Line 16 describes the points chosen from the last front, while line 17 normalizes all objectives and creates a reference set Y^s. In line 18, each solution x, that belongs to X_t, is associated with a reference point. A similar concept of clustering is used here only concerning reference points. Note that all solutions $\{FR_1, FR_2, \ldots, FR_M\}$ in X_t are related. In line 19, the niche count, which defines the number of solutions associated with the reference line, is computed. Finally, in line 20, K solutions are chosen from the last front one at a time. This implies that the solution would be copied one after the other with the help of niching techniques. This study further presents the explanation on the repair-based operator, normalization, association, and niching algorithms in the A-NSGA-III in Sections IV-B, IV-C, IV-D, and IV-E, respectively.

B. DESCRIPTION FOR THE REPAIR-BASED OPERATOR ALGORITHM IN THE A-NSGA-III

The repair-based operator mechanism is mostly problem-dependent. This study developed the repair-based operator algorithm and embedded it into the existing NSGA-III. The function of the repair-based operator algorithm is to ensure that the algorithm only searches within the feasible space and to guide the A-NSGA-III to produce unique solutions with no duplicates. The description of the repair-based operator algorithm presented Algorithm 2.

Considering Algorithm 2, lines 1 and 2 take the input and output of the algorithm, respectively. The controller position values (VA) are taken as the input, while the solution with unique value return is taken as the algorithm output VA. Line 3 of the algorithm iterates through the VA by checking the numbers of rows (i) in VA. The VA is the initial controller position set to the length of the entire dataset (20). Line 3, the algorithm iterates through VA by checking the number of columns (j) in a row in line 5. Similarly, line 4 initialises an empty list called K while line in line 6, the algorithm rounds-down values in VA to zero and converts them to an integer. Similarly, in line 7, the repair-based operator checks if K in line 4 does not have the value in line 3. Furthermore, in line 8, the algorithm assigns K to be the VA if the condition in line 7 is true, else 0 is assigned to f (line 9) if the value exists inside K. For a value within the current value of $VA[i, j]$, the algorithm checks the first number that is not in K and assigns it to q in line 10 since the value in $VA[i, j]$ exists in K up till 20. In lines 14 and 15, the algorithm checks if q is not in K. If this condition is valid, the first value that is not found in K is assigned. Consequently, line 16 assigns the value of q to K. Similarly, line 17 assigns 1 to f if value is found for K. The algorithm conditionally breaks in line 18 or 20 subject to the value of f in line 19. The reverse operation is performed from line 23 to line 27. Line 24 checks if q is not in K, then the first value found that is not in K is assigned to q in lines 25 and 26. Finally, in the same reverse order, set f as one if a value is found for K already (line 27). However, if this condition is true, the algorithm breaks in line 28 and ends.

Algorithm 2 Repair-based Operator Algorithm

1: **Input**: *VA*: *Controller position values*
2: **Output**: *VA*: *Unique solutions with no duplicate*
3: **for** *i in range* (*len* (*VA*[*i*])) : *check number of rows in VA* **do**
4: *Initialize an empty list K* : *an empty list*
5: **for** *j in range* (*len* (*VA*[*i*])) : *check the number of column in a row* **do**
6: *VA* [*i*] [*j*] = *int* (*round* (*VA* [*i*] [*j*])) : *round values inside VA to zero and convert to integer*
7: *check if K does not have the value in VA* [*i*] [*j*]
8: *Assign K* = *VA when the condition is true*
9: $f = 0$: *set* $f = 0$
10: **for** (*q in range* (*VA* [*i*] [*j*] 20)): *let q be the value between VA*[*i*][*j*] *up till* 20 **do**
11: **end for**
12: **end for**
13: **end for**
14: **if** *q not in K* : *check if the value is not found in k* (*the first one* **then**
15: *VA* [*i*] [*j*] = *q*:*Assign the first value found that is not in K*
16: *Assign the value of q to K*
17: $f = 1$: *set* $f = 1$, *if value is found for K already*
18: **break**
19: **if** $f == 0$ **then**
20: **break**
21: **end if**
22: **end if**
23: **for** (*q in reversed* (*range* (0, *VA* [*i*] [*j*])): *let q be the value between* 0 *up till VA*[*i*][*j*]/ **do**
24: **if** *q not in K* : *if the value is not found in k* (*the first one* **then**
25: *VA* [*i*] [*j*] = *q* : *Assign the first value found that is not K*
26: *Assign the value of q to K*
27: $f = 1$: *setf* = 1, *if value is found for K already*
28: **break**
29: **end if**
30: **end for**

C. DESCRIPTION FOR THE NORMALIZATION ALGORITHM IN THE PROPOSED A-NSGA-III

This section describes the normalization process of the reference points and the entire population set independent of the ranks and the corresponding algorithm in the A-NSGA-III. The normalization algorithm is presented in Algorithm 3. This algorithm normalizes all objectives between 0 and 1. This is necessary since the reference points are generated from the first quadrant on a unit hyper-plane. The algorithm also ensures that there are standard scales among the objective vectors. The algorithm normalizes both the population set and the structure reference point. The normalization algorithm is presented in Algorithm 3 while Figure 2 illustrates the normalization in the A-NSGA-III as described by [9]. The following paragraph briefly describes Algorithm 3.

In Algorithm 3 lines 3 through 6 compute the ideal point, which describes the minimum of each objective function vector computed for every objective. Line 4, the objective in line 3, is translated by subtracting the minimum value $z_{i=j}^{min}$ from

Algorithm 3 Normalize $(f^n, X_t, Y^r, Y^s/Y^a)$ *procedure*

1: **Input**:X_t, Y^s (*structured points*) *or*Y^a (*supplied points*)
2: **Output**: f^n, Y^r(*reference points on normalized hyper*$-$*plane*)
3: **for** $j = 1$ *to* M **do**
4: *Compute ideal point* : $z_{i=j}^{min} = \min_{X \in X_t} f_j(s)$
5: *Translate objectives* $f_j'(s) = f_j(s) - z_j^{min}\ \forall_s \in X_t$
6: *Determine extreme points* $(z^{jmax}\ j = 1, ..., M)$*of*$X_t$
7: **end for**
8: *Compute intercepts* a_j *for* $j = 1, ..., M$
9: *Normalize objectives* (f^n) *using* $f_i'(x) = \frac{f_i(x) - z_i^{min}}{a_i}$
10: **if** Y_a *is given* **then**
11: *Map each* Y_a *point on normalized hyper* $-$ *plane* $f_i'(x)$ *and store the points in the set* Y^r
12: **else**
13: $Y^r = Y^s$
14: **end if**

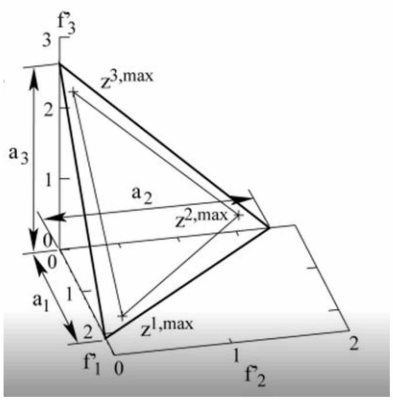

FIGURE 2. Graphical representation of the Normalization algorithm.

each objective. This converts all objectives into positive characteristics by translating the objectives into the first quadrant. This translation is necessary because the reference plane is drawn from the first quadrant, and diversity is maintained using the reference point. In lines 6 through 8, the extreme points are computed because the extreme solutions are not intercepting the $\{f_1, f_2, f_3\}$ axis. These interceptions are computed by extending the plane and giving the intersection point on objectives (1), (2), and (3). The objective function is normalized by dividing the translated objective $f_i(x)$ by the intercept in line 9. Finally, lines 10 through 14 ensure that the reference points and the population size lie on the same plane.

D. DESCRIPTION FOR THE ASSOCIATION ALGORITHM IN THE PROPOSED A-NSGA-III

Algorithm 4 explains the association between each solution and the corresponding closest reference line. The perpendicular distance between the points and the line is calculated. This association is independent of solution rank.

The reference points and corresponding front solutions are the inputs, while the outputs are the reference lines with related minimum solutions and minimum distance. In lines 3 and 4, the corresponding reference line is computed for every available reference point. In lines, 6 through 8, the

Algorithm 4 Association (X_t, Y^r) *procedure*

1: **Input**: Y^r, X_t
2: **Output**: $\pi(x \in X_t), d(x \in X_t)$
3: **for** *each reference point* $y \in Y^r$ **do**
4: *Calculate reference line* $w = y$
5: **end for**
6: **for** *each* $x \in X_t$ **do**
7: **for** *each* $w \in Y_r$ **do**
8: *Evaluate* $d^{\perp}(x, w) = ||(x - w^T xw/||w||^2)||$
9: **end for**
10: *Set* $\pi(s) = w$: $\text{argmax}_{w \in Z^r} d^{\perp}(x, w)$
11: *Set* $d(x) = d^{\perp}(x, \pi(x))$
12: **end for**

Algorithm 5 Niching $(K, \rho_{ch}, \pi, d, Y^r, FR_l, P_{t+1})$

1: **Input**:$K, \rho_{ch}, \pi(x \in X_t), d(x \in X_t), Y^r, FR_l$
2: **Output**: P_{t+1}
3: $k = 1$
4: **while** $k \leq K$ **do**
5: $CH_{min} = ch$: $\text{argmin}_{ch \in Y^r} \rho_{ch}$
6: $CH' = random(CH_{min})$
7: $I_{CH'} = x{:}\pi(x) = CH', x \in FR_l$
8: **if** $I_{CH'} \neq \emptyset$ **then**
9: **if** $\rho_{ch'} = 0$ **then**
10: $P_{t+1} = P_{t+1} \bigcup \left(x{:}\,\text{argmin}_{X \in I_{CH'}}\, d(x)\right)$
11: **else**
12: $P_{t+1} = P_{t+1} \bigcup random(I_{CH'})$
13: **end if**
14: $\rho_{ch'} = \rho_{ch'} + 1, FR_l = FR_l/s$
15: $k = k + 1$
16: **else**
17: $Y^r = Y^r/j'$
18: **end if**
19: **end while**

perpendicular distance between the point and the line is calculated for every solution in X_t and every reference line. Furthermore, the reference line showing the minimum value of the solution is computed. Finally, in lines 10 and 11, the reference line to which solution is the closest and its distance is stored as $\pi(s)$ and $d(x)$.

E. DESCRIPTION FOR THE NICHING TECHNIQUE IN THE A-NSGA-III

This subsection explains the niching technique used in the A-NSGA-III in this study. The corresponding algorithm is presented in Algorithm 5. The Niching technique selects the solutions from the last front associated with the reference line.

Three niching cases could arise from the last fronts solutions [9]. The **Case 1** explains when there exists one solution that is linked with the reference line, the **Case 2** explains when there exists no solution that is associated with the reference line and the **Case 3** explains when there is more than one solution that is linked with the reference line. Considering Algorithm 5, lines 3 and 4 reveal the solution from the last front that is copied one by one until the population is complete. The reference line with a minimum

value of niche count is identified, and one reference line is chosen at random in lines 5 and 6. Line 7 runs a check to determine the solution in the last front linked with the chosen reference line in line 6. Line 15 conditionally removes the reference line if there is no solution related to the last front. Hence, lines 8 through 19 affirm if the solution from the last front (connected with the reference line) is empty. Meanwhile, if there is a solution from the last front that is linked with the reference line (that is, there is/are the solution(s)), then the algorithm checks if the niche count of the reference line is not zero (0) in line 9. If this condition is true, the algorithm goes to line 12, to select any random solution from the last front and include it in the next generation. In line 14, the reference line's niche count increases with one (1) and removes the previously selected solution from the last front. Hence, the counter increases with one in line 15 to conduct the next niching exercise.

However, when multiple solutions from the last front are linked with the reference line and the niche count is zero, the solution closest to the reference line (line 10) would be selected to ensure diversity. Contrarily, when multiple solutions from the last front are linked with the reference line and the niche count is not zero. Any random solution could be selected from the target reference line (line 12). This implies that one solution from front 1 or front 2 is already linked with the reference line, and the diversity is already preserved. It is important to note that any randomly selected solution with high proximity from the reference line would not increase the search performance.

F. PERCENTAGE COEFFICIENT OF VARIATION (PCV)

This study uses the Percentage Coefficient of Variation (PCV) as the statistical tool to measure the diversification of the solutions across the Pareto Front concerning the average of the objective function independent of the unit of measurement [26], [27]. The [28] applied the PCV tool to dominance and diversity analysis, while [29] applied PCV to the comparison of software. Following the work of [29], the PCV is defined as the ratio of the Standard Deviation to the Average of the Objective Function. The Standard Deviation and the Average of the Objective Function are computed from the Distributed Evolutionary Algorithms in Python (DEAP) library. The DEAP library is used in this research to compile statistics on what is going on in the optimization [30]. The diversification characteristics are directly proportional to the PCV. This implies that the higher the PCV, the better the diversification characteristics [31]. This conclusion will be used to interpret the diversity value obtained in this study.

G. PERCENTAGE DIFFERENCE (% DIFF.)

This is a statistical tool that is used to express the difference (in percentage and as a fraction of the whole) between the characteristics of two items simultaneously. The percentage difference is used in this study to express the difference between the diversity of the A-NSGA-III and each of NSGA-II and MOPSO algorithms.The [32] expressed Percentage Difference (% Diff.) as

$$\%\ Diff. = \frac{Difference\ between\ two\ items}{mean\ of\ the\ two\ items} * 100. \quad (7)$$

H. PARALLEL COORDINATE PLOT (PCP)

The scatter plot in 2-Dimensional (2-D) or 3-Dimensional (3-D) has been the primary tool to observe solution vectors in the evolutionary algorithm. It helps to understand the shape, quality, and distribution of a non-domination set of solutions and the relationship that connects various objectives. However, the 2-D and 3-D plots may be associated with difficulty understanding when the number of objectives is more than three. A better alternative to studying the solution sets in this condition is to employ a Parallel Coordinate Plot (PCP). This plot displays multi-dimensional data in a 2-D graph with each aspect of the primary data converted onto a vertical axis. A parallel coordinate plot is a visualisation tool that has received modest attention in the evolutionary many-objective optimization method [33]. Sequel to this advancement, this study employed PCP to visualise and analyse the quality of the solution set obtained by the A-NSGA-III and the other two compared algorithms.

Parallel coordinates plot helps in understanding high-dimensional datasets in a simpler and better way. The PCP is utilised in order to understand the six objectives' behaviour. The parallel coordinate plot is easy to build, and it is known to scale well with the dimensionality of the dataset. Literature has established that, most often, the quality of non-dominated set solutions in both multi/many-objective evolutionary algorithms are reviewed through four measures, namely coverage, convergence, divergence, and uniformity. The focus of this study is both convergence and diversity. The convergence of a solution measures how solutions approach each other, meaning how close the solutions are to the true Pareto front. Meanwhile, divergence refers to the separation of one solution from one another. This means that the solutions are separated from each other. The primary goal of any evolutionary algorithm is to have a convergence solution and ensure diversity is preserved among the solutions.

I. HYPERVOLUME PERFORMANCE INDICATOR

Hypervolume performance indicators has gained massive attention in the literature [34]. This performance metric computes the volume of objective space dominated by the Pareto set solution. The convergence and diversity measures can be captured in a single scalar produced by a hypervolume performance metric [35]. The hypervolume indicator used the reference point to select the optimum solution. This characteristic made the hypervolume indicator preferable over other indicators that require the availability of the true Pareto Front, which is unknown in the SD-WAN controller placement. Sequel to the advantages above, this study employed a hypervolume indicator as a performance metric to assess the quality of the adapted NSGA-III compared with the two other algorithms. The Hypervolume indicator ranged between the

scale of zero (0) and one (1) inclusively. The closer the hypervolume indicator to one, the higher the quality of the algorithm performance [35]. Contrarily, the closer the hypervolume indicator to zero, the lower the performance quality. This study used hypervolume performance indicators to measure the quality of the adapted NSGA-III algorithm compared to the two most well known, similar SD-WAN controller placement Algorithms, NSGA-II and MOPSO. The NSGA-II and MOPSO algorithms were used because of their similarity with the adapted NSGA-III. These algorithms were assessed based on convergence and diversification criteria. The result and the associated discussion of the experiment are presented below.

V. EXPERIMENTATION

This experimentation was executed with a Personal Machine characterised with Intel Core i7-6820HQ CPU @2.70GHz of 1600MHz DDR3 memory, 64GB and a Microsoft Windows 10 Professional Edition specification. Python programming language was used for the experiment. A Jupyter notebook version 6.3.0. was used to compile the code. The code can be assessed as free and open-source code on the GitHub code repository with the link https://git.io/JMZNB.

The Multi-objective optimization library in python (pymoo version 0.5.0.) was used in the computation of convergence using Hypervolume performance indicator [36]. This section presents a case study to demonstrate the A-NSGA-III. The main purpose of this study is to find a suitable placement of controllers of size $k = 5$ for Internet2 OS3E topology [37] such that several conflicting objectives are simultaneously optimized. These six study objectives are average switch-to-controller latency, maximum switch-to-controller latency, average inter-controller latency, maximum inter-controller latency, load balancing, and resilience. This topology (BtEurope) [37] included 21 nodes. The following variables were the parameters set for the A-NSGA-III. The population size was 495, the number of objectives was 6, the number of dimensions was 5, the number of division (P) was 8, the reference point H was 495, the crossover probability was 1.0, and the mutation probability was 1.0 divided by the number of dimensions. According to the dataset, the lower and upper bound limits were set to zero (0) and twenty (20).

Following the founding work of [38] and the works of [39], this study started the generation number of the experiment at 100 and was systematically increased. It was observed that the experiment used 75% of the hardware computational resources, and there was no new optimal solution observed at the 500 generation number. For this study, the evaluation network scenario was extracted from the Internet2 OS3E topology in the work of [37].

VI. RESULTS AND DISCUSSION

This section presents the summarised outcome of the experimentation. The analyses results for A-NSGA-III are compared with NSGA-II and MOPSO algorithm results. The experiment results are shown in Figures 3 through 19.

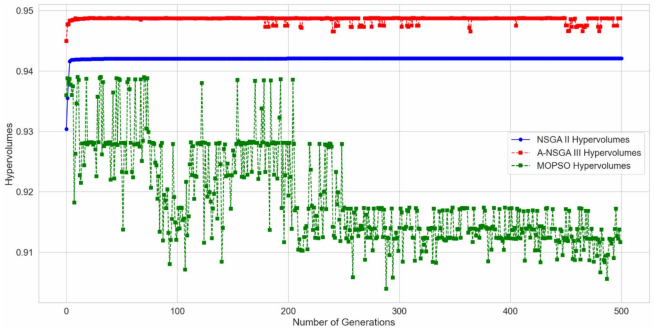

FIGURE 3. Hypervolume Indicator for the three Algorithms.

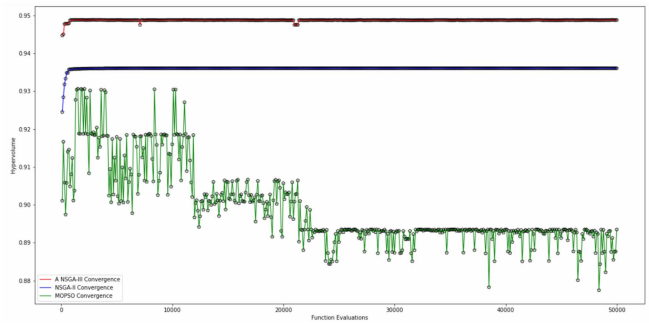

FIGURE 4. Convergence Graph of the three Algorithms.

This study aims to find the optimum position to place the controller in SD-WAN in the presence of several conflicting objectives to satisfy various network requirements. The experiment exploited six (6) characteristics of the controller placement problem in SD-WAN to realize the convergence and diversification results using Hypervolume performance indicator. These characteristics were average switch-to-controller latency, maximum switch-to-controller latency, average inter-controller latency, maximum inter-controller latency, load balancing, and resilience. This study makes use of a performance metric tool known as the Hypervolume indicator to assess the quality of the proposed A-NSGA-III, NSGA-II, and MOPSO algorithms. The results of the experimentation are interpreted and discussed in the subsequent subsections.

A. HYPERVOLUME ANALYSIS RESULTS

Figure 3 shows the merged graph of Figures 5 through 7 for the holistic comparison of the hypervolume indicators. Figure 4 shows the convergence graph of the three algorithms. Figures 5 through 7 show the descriptive chart of the hypervolume indicator for the A-NSGA-III, NSGA-II, and MOPSO algorithms, respectively. The graph shows that the A-NSGA-III had the highest (most closer to 1 among the three algorithms) traceable hypervolume indicator value of 0.94876. The NSGA-II had the traceable hypervolume indicator of 0.94314, while the MOPSO algorithm had the lowest (least closer to 1 among the three indicators) traceable hypervolume indicator value of 0.91168. The hypervolume improvement of NSGA-III (0.9488) over

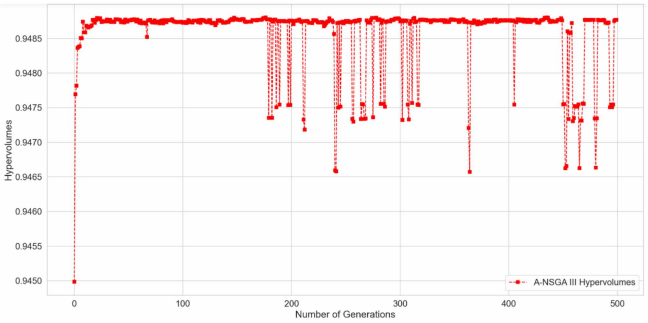

FIGURE 5. Hypervolume Indicator for the A-NSGA-III Algorithms.

TABLE 1. Diversity measurement using standard deviation and coefficient of variation.

		A-NSGA-III			NSGA-II			MOPSO		
SN	OF	AOF	SD	PCV (%)	AOF	SD	PCV (%)	AOF	SD	PCV (%)
1	0	0.2928	0.1079	36.8670	0.3651	0.0818	22.4033	0.4856	0.0312	6.4200
2	1	0.4067	0.0769	18.9024	0.4704	0.0633	13.4573	0.5078	0.0727	14.3162
3	2	0.6823	0.1613	23.6440	0.5607	0.1161	20.7105	0.3870	0.1128	29.1490
4	3	0.6271	0.2402	38.3098	0.6738	0.1554	23.0668	0.9022	0.0454	5.0297
5	4	0.0267	0.0066	24.8801	0.0318	0.0066	20.7946	0.1718	0.0201	11.7147
6	5	0.1068	0.0266	24.8808	0.1271	0.0264	20.7965	0.0430	0.0050	11.7140
		Percentage Total		167.4841			121.229			78.3436
		Percentage Difference					32.04%			72.52%

OF: Objective Function AOF: Average of Objective Function SD: Standard Deviation
PCV: Percentage Coefficient of Variation

NSGA-II (0.9431) and MOPSO (0.9117) is considered to be significant because it lies between 0 and 1. This follows the fundamental principle established by [40] and implemented by [35] and [41], [42]. The same interpretation for Figure 3 is applicable for Figures 5 through 7. Similarly, for the convergence, the A-NSGA-III had the highest (most closer to 1 among the three algorithms) traceable hypervolume indicator value of 0.94876. The NSGA-II had the traceable hypervolume indicator of 0.93646, while the MOPSO algorithm had the lowest (least closer to 1 among the three indicators) traceable hypervolume indicator value of 0.89348. Meanwhile, the number of generations (500) in the convergence algorithm was multiplied by 100 due to the internal library computations. These results imply that the A-NSGA-III has the highest convergence and diversity while the MOPSO algorithm has the least convergence and diversity. Hence, only the A-NSGA-III shows the highest convergence and diversity characteristics among the three algorithms.

B. CONVERGENCE ANALYSIS RESULTS

Figure 4 shows the convergence graph of the three algorithms. Sequel to the work of [38], the point when no new optimal solution is observed determines the point of convergence for the algorithm. In this experiment, the generation number was initially set to 100 and systematically increased until the 500 generation number was reached, and no new optimal solution was found. This implies that the algorithm converges at 500 generations. To assess further the quality of this convergence, this study employs the use of the Hypervolume performance metric to reveal further the quality of this convergence which is exhibited in Figure 4. The A-NSGA-III had the highest (most closer to 1 among the three algorithms) traceable hypervolume indicator value of 0.94876. The NSGA-II had the traceable hypervolume indicator of 0.93646, while the MOPSO algorithm had the lowest (least closer to 1 among the three indicators) traceable hypervolume indicator value of 0.89348. Meanwhile, the number of generations (500) in the convergence algorithm was multiplied by 100 due to the internal library computations. These results imply that the A-NSGA-III has the highest convergence while the MOPSO algorithm has the least convergence. Hence, only the adapted NSGA-III shows the highest convergence characteristics among the three algorithms.

Furthermore, Table 1 shows the inferential statistics of the standard deviation and the PCV associated with the six objectives for the three algorithms under consideration.

C. PERCENTAGE OF COEFFICIENT ANALYSIS RESULTS

The PCV [28] was observed to significantly reveal the internal variability than the standard deviation tool does. Hence, the PCV results were used to provide additional interpretation about the diversification characteristics of the six objectives in each of the three considered algorithms. The diversification characteristics are directly proportional to the PCV. This implies that the higher the PCV, the better the diversification characteristics [31]. Table 1 reveals that the adapted NSGA-III has the highest PCV over the NSGA-II for the six objectives. Similarly, the adapted NSGA-III has the highest PCV over the MOPSO algorithm for five objectives except for objective 2. This implies that the adapted NSGA-III has the most increased overall diversification over NSGA-II and MOPSO algorithms.

Table 1 reveals that the A-NSGA-III has the corresponding highest PCV of (36.867, 18.9024, 23.644, 38.3098, 24.8801, 24.8808) over the NSGA-II PCV of (22.4033, 13.4573, 20.7105, 23.0668, 20.7946, 20.7965) and over MOPSO algorithm PCV of (6.42, 14.3162, 29.149, 5.0297, 11.7147, 11.714) for the objective functions 0 through 5, respectively. Similarly, the PCV analyses reveals that the A-NSGA-III has a PCV total of 167.4841% over NSGA-II with the PCV total of 121.229% and the MOPSO algorithm PCV total of 78.3436%.

D. PERCENTAGE DIFFERENCE ANALYSIS RESULTS

Finding the percentage difference (% Diff.) between the proposed A-NSGA-III and the reviewed algorithms (NSGA-II and MOPSO) reveals that the A-NSGA-III outperforms both the NSGA-II and MOPSO with 32.04% and 72.52%, respectively. This result validates the established conclusion that A-NSGA-III performs efficiently over NSGA-II and MOPSO in terms of diversification when the number of objectives is more than 3. This confirms that the proposed A-NSGA-III is scalable over the NSGA-II and the MOPSO algorithms.

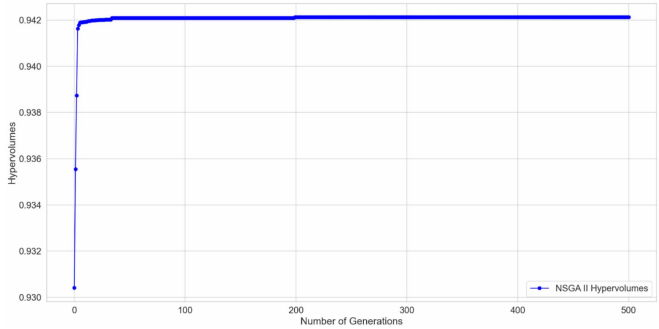

FIGURE 6. Hypervolume Indicator for the NSGA-II Algorithms.

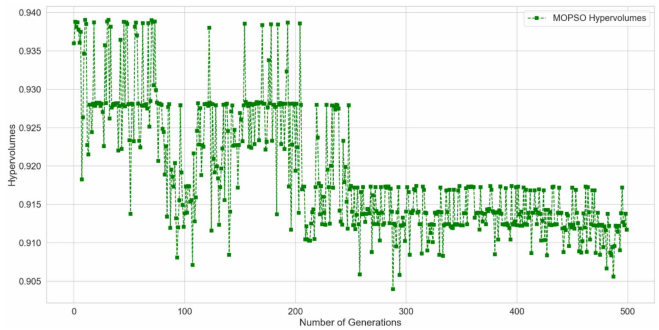

FIGURE 7. Hypervolume Indicator for the MOPSO Algorithms.

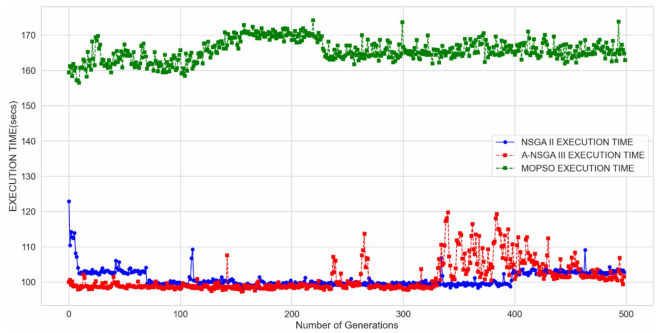

FIGURE 8. Execution time for the three Algorithms.

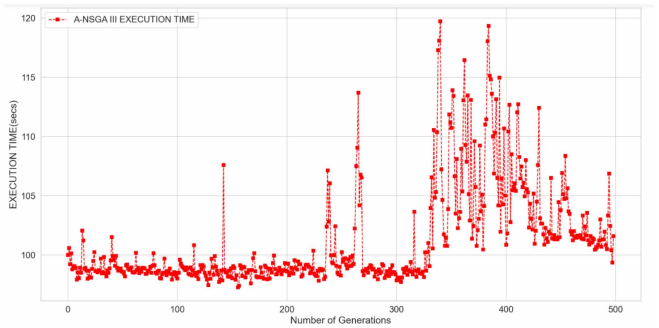

FIGURE 9. Execution time for the A-NSGA-III Algorithm.

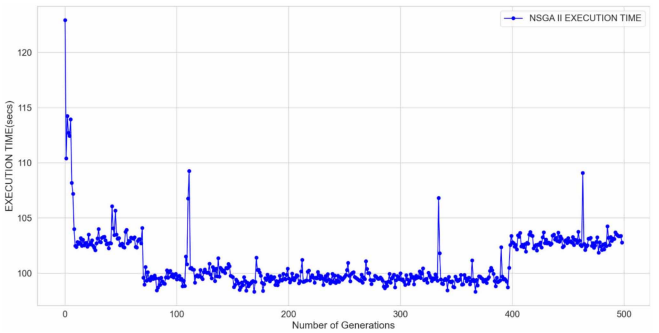

FIGURE 10. Execution time for the NSGA-II Algorithm.

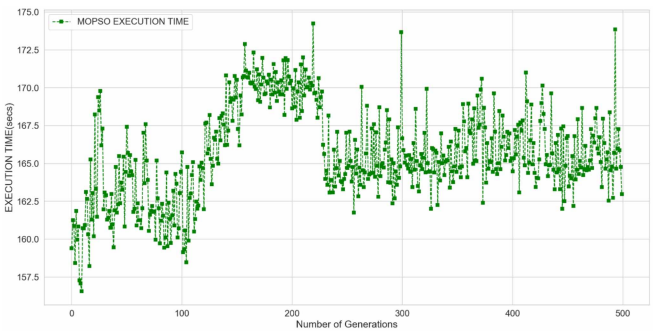

FIGURE 11. Execution time for the MOPSO Algorithm.

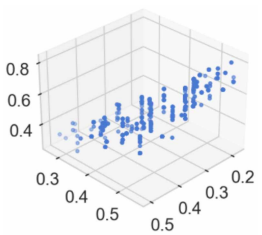

FIGURE 12. 3-D Scatter plots for A-NSGA-III Pareto sets.

E. EXPERIMENT EXECUTION TIME RESULTS

Similarly, Figure 8 shows the concatenated descriptive characteristic of the execution time (in seconds) for the three algorithms, while Figures 9 through 11 exhibit the descriptive characteristics of the execution time (in seconds) associated with each of the three algorithms. The maximum generation time was set to be 500 for each algorithm.

It was observed that, among the three algorithms, the adapted NSGA-III had a similar execution time to NSGA-II, with an average execution time of 100.961 seconds and 100.766 seconds, respectively. Meanwhile, the MOPSO algorithm had an average execution time of 165.652 seconds.

F. PARALLEL COORDINATE PLOT (PCP) RESULT

Figures 12 through 14 display the non-dominated solutions attained by the three algorithms in three-dimensional space. Analyses revealed that the A-NSGA-III is more diversified among the non-dominated solution because its solution is well spread across the objective space compared with NSGA-II and MOPSO algorithms. The NSGA-II and MOPSO solutions are not well distributed across the objective space but clustered together at most points of the entire space. NSGA-II and MOPSO algorithms exhibited these clustered characteristics revealing that such algorithms are

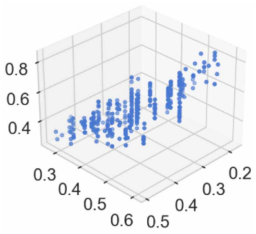

FIGURE 13. 3-D Scatter plots for NSGA-II Pareto sets.

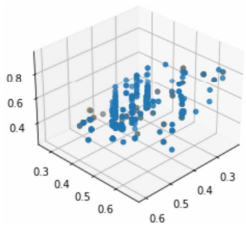

FIGURE 14. 3-D Scatter plots for MOPSO Pareto sets.

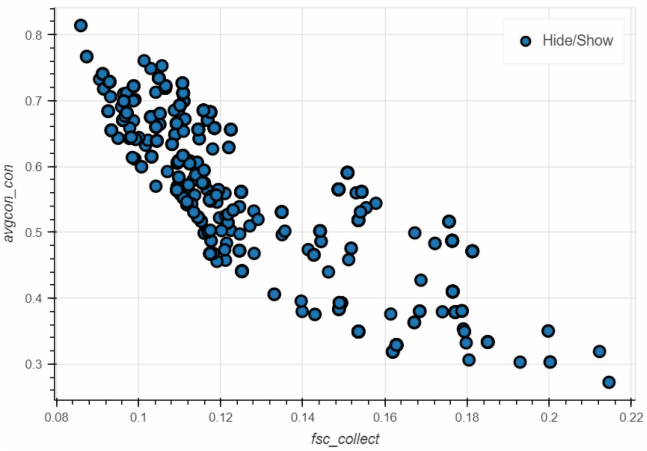

FIGURE 15. 2-D Scatter plots for A-NSGA-III.

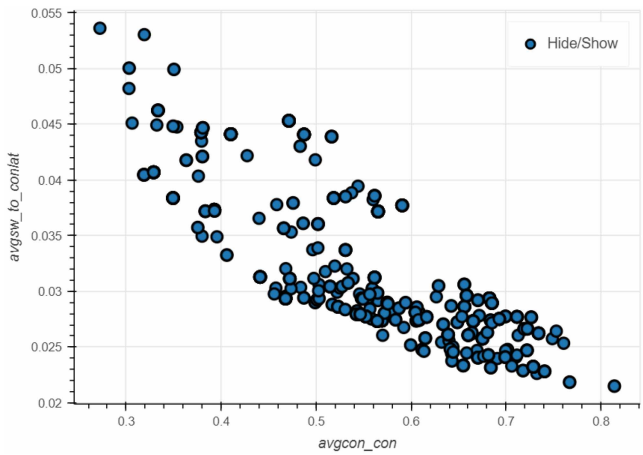

FIGURE 16. 2-D Scatter plots for A-NSGA-III.

not well distributed across the Pareto Fronts. Meanwhile, Figures 15 and 16 depict the non-dominated set solution obtained by A-NSGA-III in a two-dimensional space. These

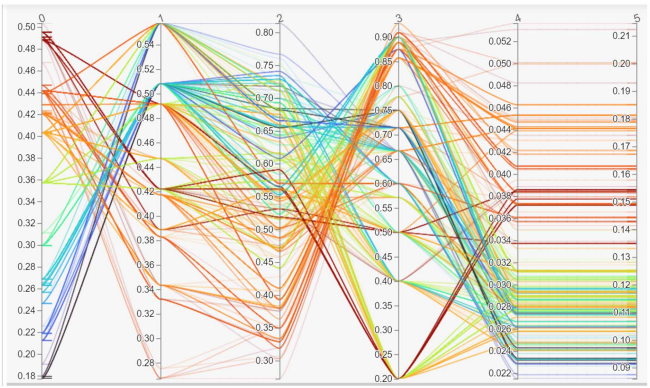

FIGURE 17. Parallel coordinate plot for A-NSGA-III solution.

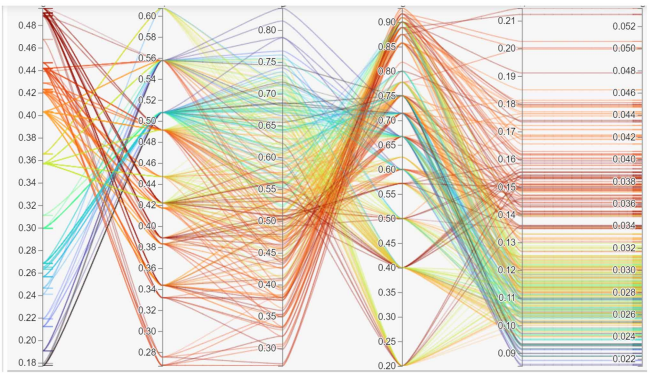

FIGURE 18. Parallel coordinate plot for NSGA-II solution.

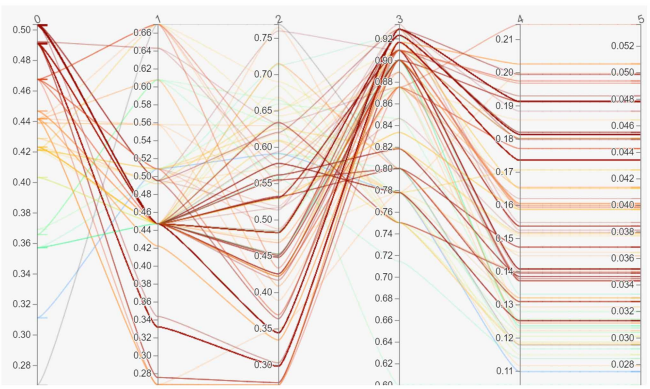

FIGURE 19. Parallel coordinate plot for MOPSO solution.

reveal that the final solutions provided by A-NSGA-III were widely distributed across the Pareto Front. The result complemented the results obtained in the previous results.

Figures 17 through 19 show the PCP of the three algorithms. It was observed that the three algorithms converged to the true Pareto front. Meanwhile, the proposed A-NSGA-III preserved the diversity of solutions extended into the border, with its solution not being clustered in one position. Contrarily, it was observed that solutions from the NSGA-II, starting from objective 1 through 6, were all huddled together, and there was no evidence of diversification across

the objective space. This result indicates that NSGA-II does not maintain a good diversity among the solutions. Like NSGA-II, the MOPSO algorithm does not maintain a good convergence towards the true Pareto front, as revealed by the PCP between 0.28 and 0.42. The higher the values, the lower the convergence with the MOPSO algorithm. However, this was contrary to what was observed in the A-NSGA-III and NSGA-II. It was also observed that the MOPSO algorithm struggled to cover the problem frontier on some objectives. Meanwhile, the solutions of the A-NSGA-III and NSGA-II algorithms appeared to have a good convergence over the entire Pareto front. These results also complimented the evidence established with the hypervolume performance indicators.

VII. CONCLUSION

Achieving an optimal controller placement in the SD-WAN environment, in the presence of more than three conflicting objectives, has been confirmed to be associated with the challenge of scalability. There is a need to develop an algorithm that can simultaneously handle several conflicting objectives for organisations to optimise more than three objectives simultaneously. This study had proposed an A-NSGA-III evolutionary algorithm from the existing NSGA-III in the Mechanical Engineering discipline. A repair-based mechanism was developed and introduced into the existing NSGA-III [9] to ensure that infeasible solutions were removed during the recombination process and also to avoid the production of duplicates among the final non-dominated set solutions. The proposed A-NSGA-III algorithm was subjected to efficiency evaluation and comparison with the reviewed NSGA-II and MOPSO algorithms, based on the diversity characteristic among the non-dominated set of solutions with six objectives. The experimentation results revealed even when the algorithms efficiently converged in the presence of more than three objectives, the adapted NSGA-III showed more convergence quality in the hypervolume performance indicator. The A-NSGA-III algorithm outperformed the NSGA-II and MOPSO algorithms in the presence of more than three objective functions. The algorithm performance to solve scalability challenge was measured with hypervolume performance metric which revealed the convergence and diversification of the algorithms performance. The diversification was measured with the hypervolume performance metric, the percentage coefficient of variation and the percentage difference metrics. The NSGA-II and MOPSO were confirmed to be characterised with the inability to obtain diverse solutions among the non-dominated solutions. Consequently, the A-NSGA-III performed efficiently over NSGA-II and MOPSO algorithms when the number of objective function is more than three. Hence, the proposed A-NSGA-III was confirmed to solve the challenge of scalability in the controller placement in the SD-WAN. This study recommends the use of the proposed A-NSGA-III algorithm for an efficient controller placement in the SD-WAN when there are more than three conflicting objective functions.

ACKNOWLEDGMENT

The authors of this study appreciate the significant contribution of Prof. Deepak Sharma of the Department of Mechanical Engineering Indian Institute of Technology Guwahati in illustrating and demonstrating the existing NSGA-III.

REFERENCES

[1] L. Mamushiane, J. Mwangama, and A. A. Lysko, *Controller Placement Optimization for Software Defined Wide Area Networks (SDWAN)*, ITU, Geneva, Switzerland, 2021.
[2] A. Sallahi and M. St-Hilaire, "Expansion model for the controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 21, no. 2, pp. 274–277, Feb. 2017.
[3] A. Ksentini, M. Bagaa, and T. Taleb, "On using SDN in 5G: The controller placement problem," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2016, pp. 1–6.
[4] O. Adekoya, A. Aneiba, and M. Patwary, "An improved switch migration decision algorithm for SDN load balancing," *IEEE Open J. Commun. Soc.*, vol. 1, pp. 1602–1613, 2020.
[5] V. Ahmadi and M. Khorramizadeh, "An adaptive heuristic for multi-objective controller placement in software-defined networks," *Comput. Electr. Eng.*, vol. 66, pp. 204–228, Feb. 2018.
[6] J. Liao, H. Sun, J. Wang, Q. Qi, K. Li, and T. Li, "Density cluster based approach for controller placement problem in large-scale software defined networkings," *Comput. Netw.*, vol. 112, pp. 24–35, Jan. 2017.
[7] D. Hock, M. Hartmann, S. Gebert, T. Zinner, and P. Tran-Gia, "Poco-PLC: Enabling dynamic Pareto-optimal resilient controller placement in SDN networks," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, 2014, pp. 115–116.
[8] A. Jalili, M. Keshtgari, and R. Akbari, "Optimal controller placement in large scale software defined networks based on modified NSGA-II," *Appl. Intell.*, vol. 48, no. 9, pp. 2809–2823, 2018.
[9] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 577–601, Aug. 2014.
[10] C. Gao, H. Wang, F. Zhu, L. Zhai, and S. Yi, "A particle swarm optimization algorithm for controller placement problem in software defined network," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.*, 2015, pp. 44–54.
[11] G. G. Yen and H. Lu, "Dynamic multiobjective evolutionary algorithm: Adaptive cell-based rank and density estimation," *IEEE Trans. Evol. Comput.*, vol. 7, no. 3, pp. 253–274, Jun. 2003.
[12] M. Hamdy, M. Palonen, and A. Hasan, "Implementation of Pareto-archive NSGA-II algorithms to a nearly-zero-energy building optimisation problem," in *Proc. Build. Simul. Optim. Conf.*, 2012, pp. 181–187.
[13] H. Li, K. Deb, Q. Zhang, P. N. Suganthan, and L. Chen, "Comparison between moea/d and NSGA-III on a set of novel many and multi-objective benchmark problems with challenging difficulties," *Swarm Evol. Comput.*, vol. 46, pp. 104–117, May 2019.
[14] F. Bannour, S. Souihi, and A. Mellouk, "Distributed SDN control: Survey, taxonomy, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 1, pp. 333–354, 1st Quart., 2018.
[15] A. L. Aliyu, A. Aneiba, and M. Patwary, "Secure communication between network applications and controller in software defined network," in *Proc. IEEE 18th Int. Symp. Netw. Comput. Appl. (NCA)*, 2019, pp. 1–8.
[16] B. Heller, R. Sherwood, and N. McKeown, "The controller placement problem," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 42, no. 4, pp. 473–478, 2012.
[17] G. Yao, J. Bi, Y. Li, and L. Guo, "On the capacitated controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 18, no. 8, pp. 1339–1342, Aug. 2014.
[18] D. Hock, M. Hartmann, S. Gebert, M. Jarschel, T. Zinner, and P. Tran-Gia, "Pareto-optimal resilient controller placement in SDN-based core networks," in *Proc. IEEE 25th Int. Teletraffic Congr. (ITC)*, 2013, pp. 1–9.

[19] M. F. Bari *et al.*, "Dynamic controller provisioning in software defined networks," in *Proc. IEEE 9th Int. Conf. Netw. Serv. Manag. (CNSM)*, 2013, pp. 18–25.
[20] S. Lange *et al.*, "Heuristic approaches to the controller placement problem in large scale SDN networks," *IEEE Trans. Netw. Service Manag.*, vol. 12, no. 1, pp. 4–17, Mar. 2015.
[21] S. Kukkonen and K. Deb, "Improved pruning of non-dominated solutions based on crowding distance for bi-objective optimization problems," in *Proc. IEEE Int. Conf. Evol. Comput.*, 2006, pp. 1179–1186.
[22] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Trans. Evol. Comput.*, vol. 18, no. 4, pp. 602–622, Aug. 2014.
[23] H. Seada and K. Deb, "U-NSGA-III: A unified evolutionary algorithm for single, multiple, and many-objective optimization," Dept. Electr. Comput. Eng., Michigan State Univ., East Lansing, MI, USA, COIN Rep. 2014022, 2014.
[24] Y. Yuan, H. Xu, and B. Wang, "An improved NSGA-III procedure for evolutionary many-objective optimization," in *Proc. Annu. Conf. Genet. Evol. Comput.*, 2014, pp. 661–668.
[25] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems," *SIAM J. Optim.*, vol. 8, no. 3, pp. 631–657, 1998.
[26] M. Marcisz, "Practical application of coefficient of variation," in *Proc. 13th Int. Congr. Energy Mineral Resour. (CIERM)*, 2013, pp. 202–208.
[27] H. Abdi, "Coefficient of variation," in *Encyclopedia of Research Design*, vol. 1. Thousand Oaks, CA, USA: SAGE Publ., 2010, pp. 169–171.
[28] A. K. Thukral, R. Bhardwaj, V. Kumar, and A. Sharma, "New indices regarding the dominance and diversity of communities, derived from sample variance and standard deviation," *Heliyon*, vol. 5, no. 10, 2019, Art. no. e02606.
[29] D. A. Agunbiade, S. O. Folorunso, K.-K. A. Abdullah, and P. I. Ogunyinka, "Two-phase sampling for stratification: Application to software industry," *Ann. Comput. Sci. Ser.*, vol. 15, no. 2, pp. 56–60, 2017.
[30] F.-M. De Rainville, F.-A. Fortin, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: A python framework for evolutionary algorithms," in *Proc. 14th Annu. Conf. Companion Genet. Evol. Comput.*, 2012, pp. 85–92.
[31] J. Mwaura, A. P. Engelbrecht, and F. V. Nepomuceno, "Diversity measures for niching algorithms," *Algorithms*, vol. 14, no. 2, p. 36, 2021.
[32] T. J. Cole and D. G. Altman, "Statistics notes: What is a percentage difference?" *Brit. Med. J.*, vol. 358, p. j3663, Aug. 2017.
[33] M. Li, L. Zhen, and X. Yao, "How to read many-objective solution sets in parallel coordinates [educational forum]," *IEEE Comput. Intell. Mag.*, vol. 12, no. 4, pp. 88–100, Nov. 2017.
[34] H. Ji and C. Dai, "A simplified hypervolume-based evolutionary algorithm for many-objective optimization," *Complexity*, vol. 2020, Aug. 2020, Art. no. 8353154.
[35] K. Shang, H. Ishibuchi, L. He, and L. M. Pang, "A survey on the hypervolume indicator in evolutionary multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 25, no. 1, pp. 1–20, Feb. 2021.
[36] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89497–89509, 2020.
[37] S. Knight, H. X. Nguyen, N. Falkner, R. Bowden, and M. Roughan, "The Internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, Oct. 2011.
[38] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.
[39] Y. Tian, H. Wang, X. Zhang, and Y. Jin, "Effectiveness and efficiency of non-dominated sorting for evolutionary multi- and many-objective optimization," *Complex Intell. Syst.*, vol. 3, no. 4, pp. 247–263, 2017.
[40] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, aggregation-, and indicator-based methods in many-objective optimization," in *Proc. Int. Conf. Evol. Multi-Criterion Optim.*, 2007, pp. 742–756.
[41] D. Brockhoff, T. Friedrich, and F. Neumann, "Analyzing hypervolume indicator based algorithms," in *Proc. Int. Conf. Parallel Problem Solving Nat.*, 2008, pp. 651–660.
[42] Z. Li, X. Wang, S. Ruan, Z. Li, C. Shen, and Y. Zeng, "A modified hypervolume based expected improvement for multi-objective efficient global optimization method," *Struct. Multidiscipl. Optim.*, vol. 58, no. 5, pp. 1961–1979, 2018.

OLADIPUPO ADEKOYA (Graduate Student Member, IEEE) received the B.Sc. degree in computer science from Olabisi Onabanjo University, Nigeria, in 2007, and the M.Sc. degree in data network and security from Birmingham City University, U.K., in 2018, where he is currently pursuing the Ph.D. degree with the School of Computing Engineering and Built Environment. His research interests include SDN/NFV and artificial intelligence. He is a member of IET.

ADEL ANEIBA received the B.Sc. degree in computer science from the University of Benghazi, Libya, the M.Sc. degree in e-commerce from Staffordshire University in 2003, and the Ph.D. degree in computing in 2008. He is an Associate Professor of Internet of Things (IoT) with Birmingham City University, U.K. His research interests include IoT, computer network simulation, evaluation, optimization, and block-chain. He is a member of IET.