IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 21, 2020, accepted September 2, 2020, date of publication September 14, 2020,
date of current version September 25, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3023974

Comprehensive Review of SDN Controller Placement Strategies

BASSEY ISONG[1], (Member, IEEE), REORAPETSE RAMOLITI SAMUEL MOLOSE[1], ADNAN M. ABU-MAHFOUZ[2], (Senior Member, IEEE), AND NOSIPHO DLADLU[1]

[1]Computer Science Department, North-West University, Mafikeng 2745, South Africa
[2]Council for Scientific and Industrial Research (CSIR), Pretoria 0184, South Africa

Corresponding author: Bassey Isong (isong.bassey@ieee.org)

This work was supported in part by the FNAS Research Council at the North-West University, Mafikeng Campus, and in part by the Council for Scientific and Industrial Research, Pretoria, South Africa, through the Smart Networks collaboration initiative and IoT-Factory Program (Funded by the Department of Science and Innovation (DSI), South Africa).

ABSTRACT Software-Defined Networking (SDN) is a network paradigm introduced to overcome the inherent challenges of traditional networks. Its architecture is either deployed with a single controller or multiple controllers. While the first is not suitable for large-scale networks, the latter is confronted with a controller placement problem (CPP) in a large-scale network environment. CPP involves the challenge of deploying the optimal number of controllers within a network while meeting certain performance requirements considered conflicting in nature such as reliability, load balancing, latency, energy efficiency, and computation time. A single optimal or random placement may not be feasible in CPP and careful planning is of the essence to find an appropriate trade-off among the metrics. To achieve this, several CPP approaches have been proposed, developed, and deployed over the years, each having its unique objectives, strengths, and weaknesses. Therefore, this paper performed a comprehensive review of some of the existing approaches to identify the unique solutions offered, comprehend the different strategies and the challenges that exist as well as provide researchers with future directions aimed at improving the optimum location and allocation of controllers, in particular, for SDN application in wireless sensor network (WSN). The findings revealed several existing solutions and algorithms as well as several challenges such as the need for an efficient algorithm, attack-aware, cost-aware, and energy-aware CPP schemes while achieving a good quality of service.

INDEX TERMS Controller placement problem, SDN, reliability, latency, load balancing, energy efficiency.

I. INTRODUCTION

SDN is a networking paradigm that emerged in recent years via several initiatives and standards to ensure network flexibility, efficient utilization, cost-effectiveness, and innovation [1]–[3]. Its design principle [4] is characterized by separating the control plane from the data plane through the provision of network programmability which brings about network dynamic configuration, control, and management simplicity [1], [5], [6]. In particular, the centralized control plane hides the complexity of the network to the application developer and delegates the underlying network control tasks to the controller. The controller as the ''intelligence'' of the network, maintains an abstract view of the entire network [2], [6]. Its functions include setting configuration parameters, creating forwarding rules, or making traffic forwarding decisions. The data plane, on the other hand, deals with the task of packets forwarding following the controller's set rules. That is, the controller manages the forwarding devices such as routers, switches, sensors, etc., by providing the rules that dictate their packet handling behavior in the network [7], [8]. SDN introduction provides great advantages to network service providers, cloud datacenters, cores, edge [2], and so on. It has widespread application in several emerging architectures such as the Internet of Things (IoT), optical networks, wireless networks, mobile networks, network function virtualization (NFV), wireless sensor networks (WSN), vehicular ad-hoc networks (VANETs), etc. [4]. Moreover, several SDN protocols have been developed such as ForCES, OpenFlow, HyperFlow, etc. with ForCes and OpenFlow being the most commonly implemented southbound protocols [1]–[3]. ForCes was initially declared the most efficient and

The associate editor coordinating the review of this manuscript and approving it for publication was Cheng Qian.

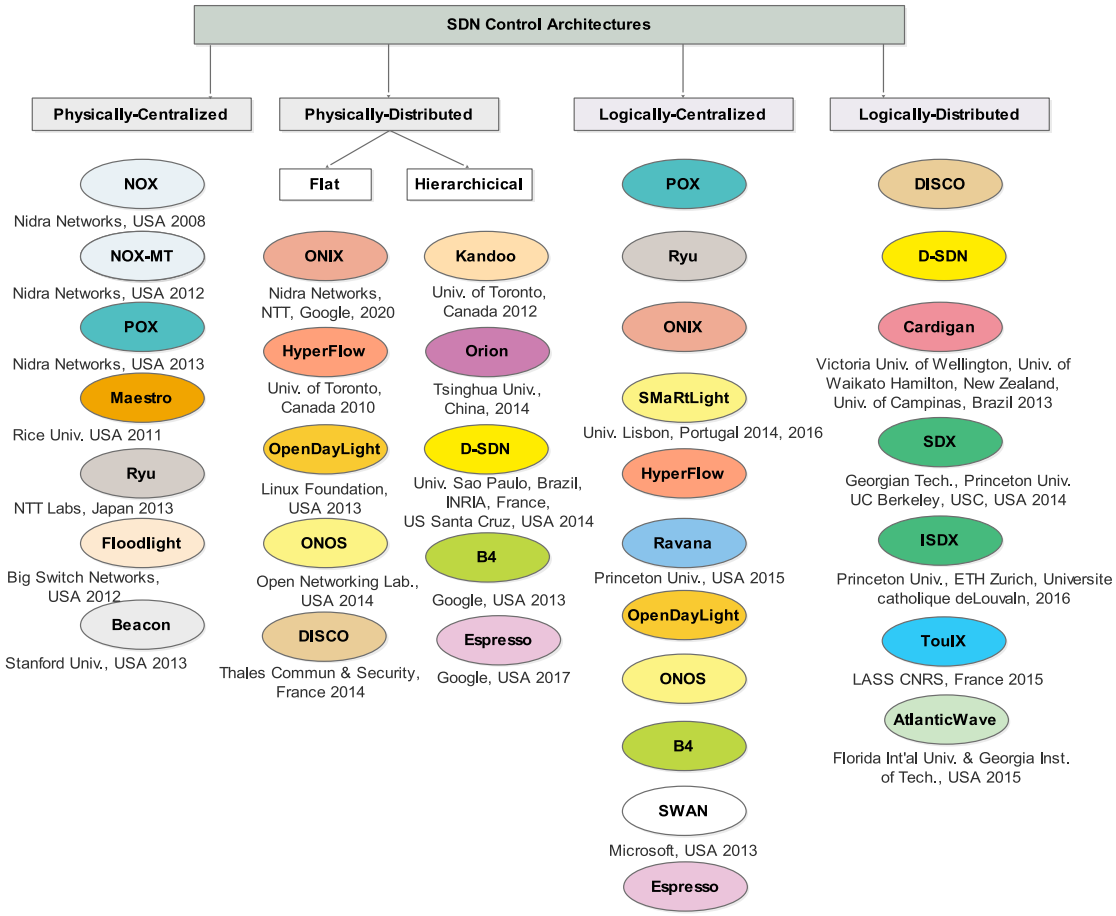

FIGURE 1. SDN control plane architectures classification [19].

dynamic southbound interface in the SDN as compared to OpenFlow. However, through advances in research, OpenFlow became the most commonly deployed protocols. The SDN southbound protocols are used to accomplish the command signals between the control plane and data plane, though each has its own design goal. Thus, this paper focuses on the control plane and the southbound interface's communication.

Currently, the architecture of the SDN control plane has been designed with either a single controller or the distributed/multiple controllers [1], [2], [9], [18]–[20]. These control architectures are further classified into physically or logically centralized and physically or logically distributed, each with several different controller types as shown in Fig. 1. [19]. Besides, each control architecture has different orientations such as flat, hierarchical, etc. However, having a single physically or logically centralized controller in charge of the forwarding nodes in a large-scale network constitutes a serious bottleneck to the network. It poses as a single point of failure that could leave the network with no intelligence, poor efficiency, unexpected high delays due to distance between controller and switches, lack of scalability support for large SDN networks, low controller processing power [1]–[3], [6], [7], [10]. These adversely impact the network performance and significantly limit the reliability, and availability of the entire networks [3], [11]. That is, any compromise on the network reliability will directly affect its availability due to failure which renders the controller or the network inaccessible. As leverage, multiple controllers have been proposed and introduced where more than one controller participate and work cooperatively in the forwarding decisions and the creation of multiple domains [1], [2], [5]. In large-scale networks with multiple domains, several controllers are placed at different locations and assigned to monitor and control switches in each domain to enhance the network quality of service (QoS), traffic engineering, routing, fast failover [1], [2], [5], [9] while ensuring that all controllers have the same global network view. Consequently, the overall network performance is significantly increased due to effective load distribution among different controllers. Also, the overall reliability increases as the failure of one controller do not inhibit the network operations, making the network more efficient and scalable [1], [5], [7], [9].

Despite these benefits, multiple controllers also bring about several challenges where optimal controller placement problem (CPP) is a typical issue [1]–[10]. For instance, to ensure network scalability, it is insufficient to merely increase the number of controllers or randomly placing the

controllers anywhere as a satisfactory performance can't be achieved [7]. This means multiple controllers have to be properly placed in appropriate locations to meet several requirements and this action involved network partitioning [1], [9], [12]. Nevertheless, partitioning the network into multiple control domains to achieve good network performance can introduce several challenges than anticipated in terms of reliability, load balancing, latency, computation time, etc. [1], [9]. It is against this backdrop that network engineers are faced with the CPP conundrum, prompting the search for answers to critical questions such as how to identify a minimum number of controllers, how to partition the network, where to locate each controller, and how to allocate controllers to switches and so on [1]–[3], [5], [8], [9], [12]–[14]. Seeking answers to these critical questions make multiple controllers' deployment for large-scale network management and its policies enforcement an important research area [1], [3]. Thus, CPP is aimed at finding the optimal location of the SDN controllers in a manner that achieves various defined objectives such as latency minimization, load balancing, energy efficiency, and enhanced reliability [2], [3], [13] which are critical to SDN's performance in large-scale networks. Moreover, the rationale for CPP is also based on the fact that both capital expenditure (CAPEX) and operational expenditure (OPEX) of the network are highly impacted by the numbers of the controllers to be deployed [3]. Therefore, careful planning and estimation of the optimal location-allocation of the number of controllers are critical to reaping the benefits of CPP in large-scale SDN-based network.

In recent years, WSN has gained considerable attention due to smart sensors advances that are critical to IoT. WSN consist of sensor nodes deployed over a chosen environment to monitor physical and environmental factors such as humidity, temperature, motions, vibrations, and so on [52]. However, these sensors are inherently resourced constraints such as limited energy, processing capability, and data storage and communication bandwidth [52]. These, in turn, poses great challenges to the WSN and its future for IoT. Through advances in research, SDN emerged as a potential solution to the challenges faced in the traditional networks, and its application in WSN resulted in a new network model known as software-defined WSN (SDWSN). This new networking model was designed to address the inherent challenges and complexities associated with WSN and promote flexibility, latent efficiency, innovations, and so on [52]. The challenges include energy inefficiency, network management, and configuration, scalability, security threats, routing, mobility as well as localization. In the perspective of scalability, to effectively utilize the services of the SDWSN in a large-scale setting, the CPP is equally important like the SDN. However, for the SDWSN and due to the nature of the WSN, the CPP strategy will need to involve new objectives such as nodes mobility and energy efficiency in addition to the different SDN CPP strategies. That is, albeit the SDN-based CPP techniques are also critical in the placement of SDWSN controllers, the techniques of placement may not completely address the issues involved in SDWSN. Furthermore, while some placement is achieved statically, others are realized dynamically. Hence an SDWSN based CPP will require dynamic placement and virtualization amongst a few dynamic controller placement methods could be adopted to ensure effectiveness in the CPP strategy. Consequently, Kumari and Sairam [46] suggested that CPP in SDWSN can be effective by exploring the WSN's unique topology, range, energy limitations, and other factors.

Currently, several CPP approaches have been proposed and developed over the years. Each approach has its algorithm(s), strengths, and weaknesses. Moreover, an efficient CPP algorithm is yet to be fully realized. Authors in [15], [22] argued that an efficient CPP is a time-efficient one, uses an online search algorithm, and prioritizes QoS. Though several existing works were not considered in this review, authors in [48] employed an online CPP algorithm, and several other schemes achieved improved QoS, there is a need to channel CPP in this direction. In terms of QoS, most existing schemes only considered a single objective while some considered multi-objectives but not all objectives. Also, multi-objective functions have some drawbacks as the objectives are conflicting in nature, resulting in inefficient solutions or poor QoS. Furthermore, existing CPP schemes have been majorly designed for SDN application in wide area networks (WAN) while less or no attention is paid to SDWSN and other network models, considering the key role of SDWSN in the future of IoT, for instance. Therefore, this paper performed a comprehensive review of the existing CPP approaches to identify some of the strategies applied, which network architectures applicable, challenges that exist, and also to assist future researchers with important research directions that could guide in the design of efficient and dynamic CPP strategy for the SDWSN model. To this end, the contribution of this paper is summarized as follows:

- Provide a brief explanation of CPP and its performance metrics which were commonly considered in recent CPP studies and their formulations.
- Provide a comprehensive analysis of some of the existing CPP approaches and strategies which are categorized according to their algorithm or strategy type including clustered-based approaches, linear and integer programming, machine learning (ML), evolutionary computing based, game theory approaches, and so on. Moreover, the strengths and limitations of each approach are highlighted.
- Provide discussions on the respective identified challenges and possible future research directions on the general CPP perspective.

This paper is organized as follows: Section II discusses the nature of CPP in SDN, Section III presents the summary of related works, and Section IV is the in-depth analysis of the existing CPP strategies, Sections V is the paper discussions and possible research directions, while Section VI is the paper conclusion.

II. THE CONTROLLER PLACEMENT PROBLEM

CPP is one of the critical problems faced in the realm of SDN when multiple controllers are deployed in the management of the networks as well as the enforcement of network policies in such controllers. It deals with finding the required feasible or optimal number of controllers and their location in the SDN networks to meet various performance and QoS requirements [1]. Generally, most CPP is modeled as a graph, $\mathbf{G} = (\mathbf{V}, \mathbf{E}, \mathbf{U})$ topology where $\boldsymbol{V}$ is the set of n switches, $\boldsymbol{E}$ is the set of edges (physical links) among switches or controllers and $\boldsymbol{U}$ is the set of k controllers [15]. Studies on CPP are centered on designing techniques to solve for the value of $\boldsymbol{k}$ and the relation given by $\boldsymbol{U} \rightarrow \boldsymbol{V}$, which for instance, is the shortest path latencies between each pair of nodes when minimizing latency as the only objective in the objective function [12], [15]. The formulation is an optimization problem that is either data-driven [16] or metric-driven formulated to find the minimum or maximize cost, the optimal number of controllers, switch-controller (SC) latency, controller synchronization time or hybrid metrics or multi-objective optimization problem about controller location or placement [1]. The model allows searching for the number, type, and location of controllers [1], [9] which are the key to network performance and QoS [2], [3]. CPP was first considered by Heller *et al.* [17] as a non-deterministic polynomial (NP)-hard problem and is similar to the facility location problem [2], [3]. Solving CPP is challenging and requires proper planning and good decisions making to achieve optimal location and satisfactory results [2]. For multiple controllers placement, several factors influence the SDN performance such as reliability, controller load balancing, latency, operational costs, and response time of events [1], [2], [6], [8], [10]. For instance:

A. LATENCY

Latency is one of the core factors considered in CPP. It depends highly on the distance between nodes in a network which is critical to packet propagation. Two latencies are important: propagation and processing. Propagation latency is the response time among the controllers or switches which is influenced by the distance between them and the processing latency is highly influenced by the controller's load. In other words, the optimality of the latencies influences both Controller-to-Controller (CC) and SC delays respectively. Moreover, processing latency increases as the flows from several switches increases, and whenever the SC latency exceeds a threshold, the overall latency of the network increases greatly. This, therefore, affects the network availability in terms of responding to network events or push forwarding commands to switches on time.

B. CONTROLLER LOAD BALANCING

An increase in the number of switches controlled by a controller increases the controller's load as well. If a load of a controller exceeds its processing ability, it could result in queueing delays or new request not served from the switches since it has only the capacity to manage a limited number of switches' request at the same time [1], [8]. That is, the controller fails to process requests, and the communication overhead of SC or CC increases, therefore, it is important SC assignment is well-balanced. However, finding the best placement with minimum controllers and switches assignment is a difficult task when load balancing is considered [14].

C. FAULT TOLERANCE

In the SDN, each switch is assigned to a controller and a controller's failure affects the link connecting the controller and switch since no requests the switch will be received or processed at the controller. Accordingly, link failure negatively impacts the controlled switches and restrict some functions of the control plane whenever the SC connection is lost. The switch receives no new routing instructions and all packets are dropped [8]. Thus, the best placement that minimizes the number of controllers while ensuring reliability is important.

D. OPTIMAL NUMBER OF CONTROLLERS

In this perspective, an increase in the number of switches also increases the network complexity to meet various network performance requirements. That is, having several unplanned switches assigned to a controller brings about increase complexity on the controller since it has to process several requests within a given time interval. This poses a great challenge when determining how many controllers should be deployed and their placement to increase network performance and QoS. However, several approaches such as traversal searching to find optimum performance numbers is considered time-consuming [8].

E. COST

Optimally and merely placing the controllers in a large scale network topology has a huge impact on the overall expenditure. That is, finding the optimal number of controllers to deploy involves considering all possible costs such as budget limitations, purchasing, repairing, and maintenance costs [1]. The terms CAPEX and OPEX are considered in the SDN controller placement when minimizing the total cost [27]. Thus, finding the best number of controllers to be deployed that minimizes cost is an important factor.

F. INTER–CONTROLLER COMMUNICATION

In the SDN, CC or inter-controller communication is maintained via state synchronization to achieve global consistency. In this case, controllers communicate when they wish to pass a message to switches controlled by others. Therefore, such communication impacts the performance of the end-to-end communication between different switches controlled by different controllers [8] and is a critical factor in the perspective of CPP.

In general, during CPP that involves identifying the required number of controllers and their locations, the key considerations are the performance metrics [5], [10], [12].

The metrics are used to evaluate the quality of the different controller placement in the network and are critical to its performance, QoS, efficiency, scalability, and reliability [5], [10], [12].

III. RELATED SURVEY WORKS

This section presents some of the related works which are separated into three parts: papers on different control architecture, survey papers on CPP outlining the algorithms, problem formulation, the performance metrics, few works that suggested various approaches, and models that are considered in CPP. The discussion is as follows:

Karakus and Durresi [18] conducted a comprehensive survey on SDN-based control plane scalability issues by providing categorization and taxonomy of the state-of-the-art. Categorization involved two approaches: topology and mechanism-related. The topology-related approaches involved the topology of different architectures, their relationship, and scalability concerns concerning centralized and distributed controllers design. The mechanism-related involved several mechanisms to optimize controllers and their scalability challenges such as parallel-based and routing scheme-based optimization. The authors further identified the control plane and data plane separation, controller load, and SC communication delay as the key bottlenecks to SDN scalability. Several challenges were also found and reported as well as open problems that need further investigations to ensure more scalability in the SDN such as controller placement, controllers' failure, flow setup latency, state or policy distribution or consistency and so on [18]. Similarly, Bannour *et al.* [19] also surveyed on distributed SDN control with a detailed analysis of the state-of-the-art of the distributed controllers' platforms by evaluating their merits and demerits for extensive comprehension. They provided both physical and logical classification as potential guidelines for research and deployment. Discussions were provided on several critical challenges and offered insights into developing and future research trends in distributed SDN control. Accordingly, scalability, consistency, reliability, and interoperability were identified as parts of the key challenges confronting the design of an efficient and robust distributed SDN controller platforms. Insights offered to counter the challenges include the introduction of major standardization at all levels of CC communication, effective CPP, and knowledge sharing problem. In the same vein, Zhang *et al.* [20] surveyed recent progress in multiple SDN controllers. They discussed the benefits, detailed design principles, and the architecture as well as the challenges faced by multiple controllers. Moreover, the strengths and weaknesses of several different research work on controller placement and scheduling were also discussed and analyzed. They presented some future works and research directions. Oktian *et al.* [21] also surveyed different design approaches to ensure a logically centralized view in multiple distributed controllers. They found that such approaches can be classified into different design choices among SDN adopters and each choice may impact many issues like robustness, scalability, privacy, and consistency. Moreover, they provided an analysis of each of the models in terms of their pros and cons. The findings showed each design produced some features and one can succeed in resolving one issue but may fail in another. They presented the design choice that can be used to construct a distributed controller that overcome all the above issues.

In terms of CPP, Wang *et al.* [22] surveyed state-of-the-art solutions to CPP and provided a taxonomy of the CPP based on their objectives such as minimize network latency, maximize reliability, and resilience as well as minimize the cost of deployment and energy consumption. They proposed a new approach to minimize the propagation latency between SC, suggested several types of future research and highlighted some research challenges, and open relevant issues on CPP. Some of the issues identified include the need for an efficient algorithm, multi-objective optimization, cooperation among controllers, and cost awareness. Similarly, Lu *et al.* [15] comprehensively surveyed the state-of-the-art of SDN CPP with a focus on the optimization objective. They provided discussions on CPP and classified it into four aspects such as latency, reliability, cost, and multi-objective concerning their objectives. Also provided an analysis of specific algorithms, given the methods and simulations in different application settings. Moreover, several open issues and research challenges for future work on CPP were identified and reported. Some of the issues found include the need for efficient CPP algorithm, cost-awareness, attack-awareness, and virtualized CPP. Accordingly, Singh and Srivastava [23] classified CPP into minimizing, maximizing, and multi-objective approaches. For each approach, they carry out an in-depth analysis of the solutions, their limitations, and suggested several open and future research to bring innovations and reliable CPP. Some of the identified challenges include network partitioning approach to CPP that considers load balancing and all possible failures, application dynamic clustering into CPP, implementation, and deployment large-scale SDN, QoS in function placement problem for network function virtualization, finding the advantages and disadvantages of clustering mechanisms, ILP, Greedy, Heuristics approaches of CPP as well as the efficient solution for large-scale networks with dynamic traffic load and fault. Das *et al.* [2] also comprehensively surveyed the SDN-based CPP and discussed various developments in the area. They classified CPP formulation based on several performance metrics such as latency, QoS, resilience, etc. as well as highlighted on them. Moreover, they summarized the various schemes for solving CPP and their limitations and broadly categorized them into optimal based solutions and heuristic-based suboptimal solutions. They also highlighted CPP application in a wide range of contexts such as mobile/cellular, data-center, large-scale WAN, and next-generation networks such as VANETs and 5G. Several open issues and potential research directions were also provided.

Furthermore, Kumari and Sairam [46] also performed a comprehensive review of several performance metrics and the features of the existing CPP solutions from the perspective

of SDN in terms of wired or wireless networks. They also provided several future research directions such as controller addition and deletion, controller relocating, switch migration issues, CPP for domains like IoT, sensor networks, etc. In a similar work, Hollinghurst *et al.* [24] also performed an analysis of four different algorithms used for CPP such as the k-means++, full search, local search, and linear programming (LP). They compared the scalability and the accuracy of the algorithms for the k-median problem. Experiments conducted show the scalability of controllers' placement varied considerably for algorithmic complexity. Full search algorithms were considered infeasible for large-scale networks, LP algorithms suitable for only restricted network size, while the local search and the k-means++ algorithms were considered the most scalable. In terms of accuracy, full search algorithms were found to be optimal while local search and k-means++ had varied standard deviation and reduced variance respectively. Lastly, Adebayo *et al.* [25], reviewed four CPP models with a focus on their objectives, strengths, and weaknesses. They presented an analysis of the feature selection methods employed in each model to search for the optimal number to be deployed and their location as well as their impact on the accuracy and convergence time. Findings show that the models were not suitable for real-world applications and concluded that accuracy and complexity constraints can only be met if a meta-heuristic algorithm is used rather than a selective search approach. Furthermore, albeit three of the models considered traffic conditions, they suffered high convergence time. Thus, high accuracy and low convergence time can be achieved via refinement of the feature selection approach. In order words, optimality via maximizing accuracy and minimizing convergence time. In a nutshell and according to the authors, the paper provides a stepping stone for optimized controller placement model development for SDWAN.

In the above-related works, some of the studies focused on the general issues of multiple controllers in SDN such as scalability, etc. while others focused on CPP approaches that addressed the issues in terms of the impact of different types of metrics or objectives on performance and QoS as well as problem formulation. Also, they provided research directions to address some of the issues in CPP in general. However, this paper performed and in-depth analysis of the most utilized and recent approaches to address SDN CPP. We specifically focused on the strategies and algorithms used in each case, metrics considered, metrics formulation, the strengths, and possible limitations. Most importantly, we identified various future research directions with a focus on designing a CPP scheme for the SDWSN that is efficient and dynamic. More information about metrics formulation can be found in [2], [15], [22], [23].

IV. ANALYSIS OF CPP STRATEGIES

Recently, several approaches have been proposed and developed to solve SDN-based CPP and each approach has its own set of objectives which are either to minimize or maximize or both [3]. Existing approaches in terms of controller allocation and the optimal number of controllers have been classified as shown in Fig. 2. Fig. 2 present a simple yet informative classification of CPP based on several factors or metrics that influence SDN performance and QoS as well as how such metrics are optimized [3], [15], [23]. As shown in Fig. 2, latency is considered the core influencing factor in the CPP due to the criticality of message exchanges involving SC and CC in the SDN. Albeit there are several network latencies, propagation, and processing latencies are the important latencies critical to CPP [15], [23], [60]. Moreover, in a multi-domain network setting, propagation latency is composed of two types: average latency (avg_latency) and the worst-case latency (worst_latency) [15], [58]. Both SC_avg_latency [8], [17], [31], [58], [59] and SC_worst_latency [2], [28], [34]) are considered within a given network domain while (CC_avg_latency) [35], [44], [50], [59] is considered for inter- controller communication between domains [15]. Accordingly, SC_avg_latency constitutes the average packet transmission delay between SC in the SDN and its formulation is shown in Equation 1, while SC_worst_latency is the maximum transmission delay between SC as formulated in Equation 2 [15], [23].

$$SC\_avg\_Latency = \frac{1}{n}\sum_{v\in V}\min_{u\in U} d\,(v,u) \tag{1}$$

$$SC\_worst\_Latency = \max_{v\in V}\min_{u\in U} d\,(v,u) \tag{2}$$

In the perspective of inter-controller communication, CC_avg_latency is important since network state consistency is critical to controllers' communication and network availability. Its formulation is shown in Equation 3 while processing latency (*prgn_Latency*) [56] which depends on the processing ability and the controller's load is expressed in Equation 4 where L is the latency.

$$CC\_avg\_Latency = \frac{1}{k}\sum_{u\in U}\min_{u'\in U} \quad d\left(u,u'\right) \tag{3}$$

$$prng\_Latency = \min\left(\max_{u\in U} L\,(u) - \min_{u'\in U} \mathrm{L}\left(u'\right)\right) \tag{4}$$

Moreover, reliability involves measures set in place to ensure that in the event failure, there is no single point of failure and the communication between controllers and switches is not interrupted. This is the basis of its criticality in SDN-based CPP [15], [23] and several approaches exist to improving reliability such as multiple controllers [7], [15], multiple control-path [15], [23], [36]–[38], [54], and minimizing control-path [15], [33], [39]. While the multiple control-path ensures that there exist at least two disjoint paths linking a controller and a switch, multiple controllers ensure a switch is linked to several controllers to eliminate a single point of failure as experienced with a centralized controller. Moreover, minimizing control-path involves a reduction in the physical unit of the control-path, thereby improving reliability. These approaches are based on the probability of failure,

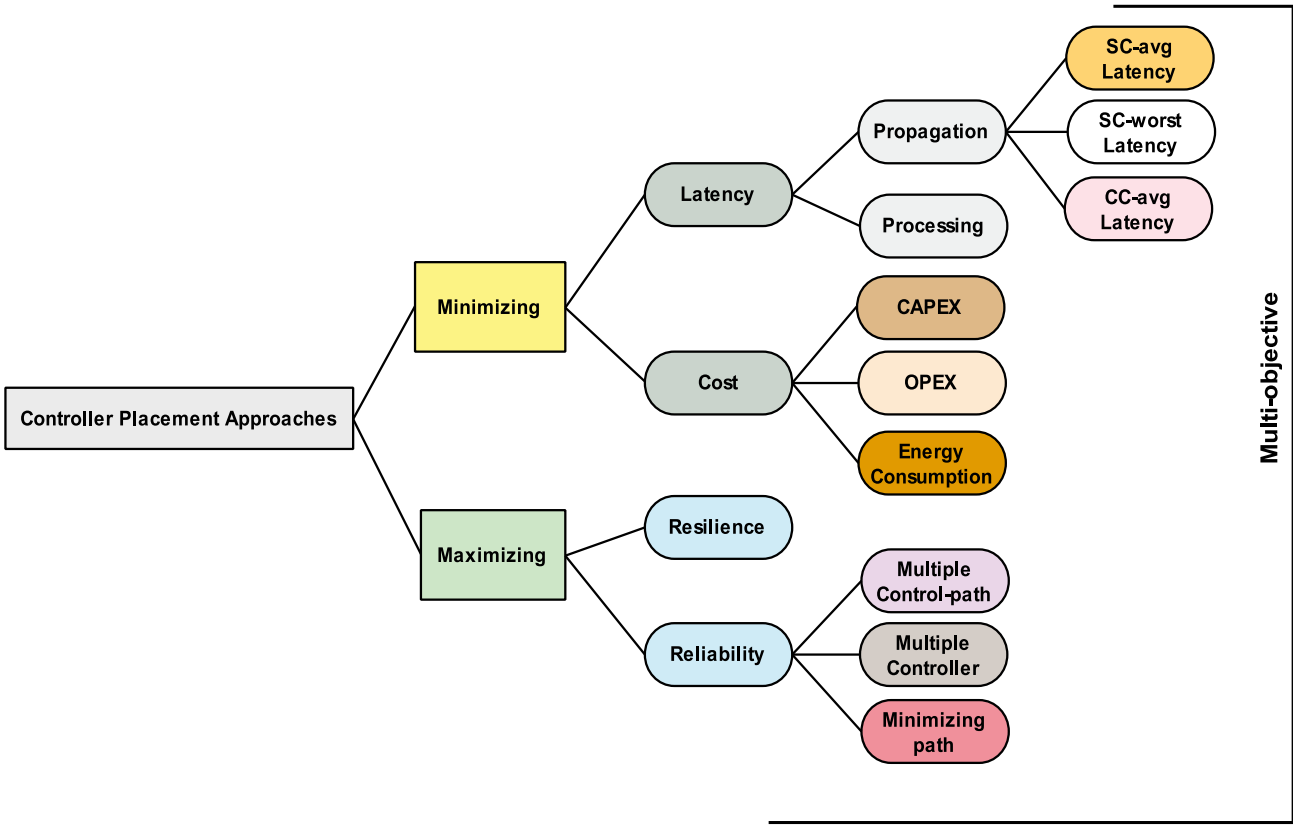

FIGURE 2. Classification of controller placement techniques [3], [15], [23].

ρ [15] and Equation 5 presents the formulation for multiple controllers where ρ(v, u) indicates control-path availability probability.

$$R\_mc = \max \sum_{v \in V} (1 - \prod_{u \in U} (1 - p(v, u))) \tag{5}$$

where ρ(v, u) is given by:

$$\rho\,(v, u) = \prod_{t \in d(v,u)} (1 - \rho t) \tag{6}$$

In the same vein, during multiple controllers' placement, an influencing factor such as the cost is considered from two perspectives: deployment [1], [40], [41] and energy consumed in the network [15], [44]. Cost in terms of deployment is tackled by adopting a location-allocation strategy that minimizes the overall cost of controllers, switches, and their running expenses given as CAPEX and OPEX [3], [23]. Equation 7 shows the problem formulation for deployment cost (D_{cost}) where C_x, C_y, and C_{cin} are the cost of controllers, controllers, and switches connection and inter-controller connection respectively [15].

$$D_{\cos t} = \min \left(C_x + C_y + C_{in}\right) \tag{7}$$

Moreover, the cost energy consumed in the network (E_{cost}) is presented in Equation 8 where σ(v) denotes the number of packets transmitted and φ(v, u) is the overhead incurred when v communicate with u [15].

$$E_{\cos t} = \sum_{v \in V} \sigma\,(v)\ \ \Phi\,(v, u) \tag{8}$$

Lastly, when multi-objectives [6], [50] are considered in the controller placement, though conflicting in nature, two or more metrics are considered. Albeit, there is no fixed number of metrics to optimize, it all depends on the researcher's objective. In this case, the problem is formulated as shown in Equation 9 [15], [23].

$$M_{ob} = \max \left[\text{Latency, Reliability, Cost, } \ldots\right] \tag{9}$$

Therefore, to satisfy each performance metric or group of metrics, several algorithms or approaches have been proposed and utilized. Some of these approaches are categorized, analyzed, and discussed as follows:

A. BRUTE-FORCE AND CLUSTERED-BASED CPP

In the realm of CPP, several approaches are based on brute-force, k-center, k-means, k-critical, and other clustering algorithms. TABLE 1 summarizes the algorithm, implementation, and metrics considered in each study.

TABLE 1. Clustered based CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[17]	k-Centre (Brute-force)	Simulations: Internet Topology Zoo (Internet2 OS3E 256 topologies)	SC propagation latency	CN
[38]	Clustering algorithm	Simulations: Internet2 OS3E (37 nodes), Internet Topology Zoo (43 nodes)	Reliability	WAN
[43]	k-means & k-Centre	Simulations: MATLAB, (15) sink nodes, (3) local controllers and the global controller, 2 GHz CPU and 1/2 GB RAM, ONOS mastership	Number of controllers, propagation latency, Reliability	WSN
[47]	Clustering-based network partition algorithm	Simulation: Matlab Internet Topology Zoo (Internet2 OS3E & ChinaNet)	end-to-end latency, the queuing latency of controllers	WAN
[8]	Density-based cluster	Simulations: Internet Topology Zoo.	Reliability, SC latency, CC latency, time consumption	CN
[54]	K-critical	Simulations: MATLAB, (90) controllers	Number of Controllers, Location, Delay, Reliability	CN (Datacenter)
[55]	Spectral Clustering Placement Algorithm	Simulation: MATLAB, Internet Topology Zoo (Internet2 OS3E)	Latency, Load balance, Reliability	WAN
[56]	K Self-adaptive	Simulations: MATLAB, Internet Topology Zoo, Internet2 OS3E	Latency, Load balance, Reliability	WAN
[58]	Algorithm based k*-means	Simulations: Internet2 OS3E	Latency, Load balance	WAN
[59]	K-median (Topological potential and Minimum-cost flow)	Simulations, Python language 4 Topology Zoo, Internet2 OS3E	Latency, Load balance	WAN

***CN = Computer Network, **WAN = Wide Area Network ***WSN = Wireless Sensors Networks**

1) K-CENTER AND K-MEANS

Heller *et al.* [17] were the first to define CPP as NP-hard and employed the k-center algorithm based on brute-force, in particular, an exhaustive algorithm to achieve how many and where controllers should be deployed. The study considered several factors to characterize the propagation latency between the switches and the controllers and explores the trade-offs between them. Simulations performed on different topologies covering diverse geographical areas found both average and worst latency not optimized in most topologies. Moreover, a controller is adequate to meet common network requirements, given the delay in communication, and adding extra controllers could reduce the latency. Also, the strategy consumed too much time in large-scale networks and utilized topologies that can't be solved within 30 hours. Fault tolerance and capacity limitation of the controllers were not considered and provide a less theoretical analysis of the algorithm.

In a similar study, Kobo *et al.* [43] proposed a CPP for SDWSN intending to optimize a distributed control system for simplicity and application in real-world network deployment. They used both k-means and k-center as well as integrated both CPP and efficient controller re-election mechanism to optimize their proposed fragmentation model for SDWSN [43] to ensure minimum latency and resiliency. The fragmentation model divides the network into different clusters with each having its control service: local and global controllers to minimize the distance between the controllers and the sink nodes. The objective was to make the network scalable, reliable, and better performance. They considered the optimal number of controllers to be deployed, propagation latency, and resiliency in the event of failure. Simulation results show the same method of optimal placement doesn't work for both local and global controllers. Thus, k-means was then used for local controller placement while the k-center was used for the global controller location. Moreover, a slight addition of delay/latency was observed as the change of controller took place. However, the latency addition had no significant impact on the network but was more when the distance was not considered. The strategy is limited by the simulation tool used which does not accommodate more than 39 sensor nodes. Jimenez *et al.* [54] also proposed a CPP approach that ensures reliability in the SDN with a strict focus on the SDN controllers. The authors emphasized the consequences of poor controller selection which negatively impacts the robustness of the network control as well as the cost and

inefficiency associated with deploying an excess number of optimal controllers. Thus, to achieve reliability, a mechanism was deployed which demonstrated that the choice of the shortest path between controllers does not improve load and robustness of the control layer and that the optimum number of controllers is determined by the network physical features [54]. A K-critical algorithm was proposed to effectively select the least number of controllers and their location to build a robust control layer, guard against failures, and load imbalance among the controllers. Moreover, the effectiveness of the approach was established by comparing it with other CPP solutions such as K-means and K-centre algorithms. The evaluation was performed using Matlab and with sparse, medium, and dense networks while considering controllers between 0 – 90 for various instances. The results showed the shortest tress has the least data loss and at least, five controllers are enough for networks considered sparse [15]. Similarly, Kuang *et al.* [58] proposed a CPP approach for SDN-based WAN that is based on network partitioning. The authors proposed an algorithm-based K*means or hierarchical k-means which is used to partition the network into several subnetwork domains controller by a single controller. They considered latency and load balancing as the performance metrics to minimize the SC_worst_latency and controllers' load. The approach was evaluated using the Internet2 OS3E topology and compared with the optimized k-means algorithm. The results obtained showed the superiority of the algorithm-based k*means in terms of propagation latency and with a better load balance. Moreover, the authors also stated the need to consider reliability in their future work.

2) OTHER CLUSTERED BASED ALGORITHMS

When considering multiple objectives with the consideration of the optimal number of nodes within a cluster, a more efficient, or optimized approach is important. In this regard, Liu *et al.* [38] CPP strategy aimed to optimize the network average reliability. It is an optimization problem which considered the shortest path between the controller and switches. It employed a clustering algorithm for optimal placement for the shortest-path and a greedy algorithm for multi-paths between controller and switches for sub-optimal placement. Moreover, a reliability factor was defined to replace the network average reliability in multi-paths which in turn reduced the computational complexity. Simulations on real network topologies showed the global clustering algorithm performed better than the local greedy algorithm in the case of multi-paths regardless of the number of controllers. This according to the authors, was due to factors such as the number of paths, path correlation, length of the path, etc. which influence the reliability of multi-paths between nodes. Accordingly, Liao *et al.* [8] also proposed a generic algorithm for SDN-based CPP which considers latency, load balancing, and link failures via the incorporation of clusters. The aim was to ensure if a controller's load exceeds a certain threshold, switches assigned to it can be automatically be reassigned to other controllers. The method employed is called the Density-Based Controller Placement (DBCP) which utilizes a density-based switch clustering algorithm (DBSCA) maintained in a table to split the network into several sub-networks based on the network architecture. In this case, the optimal global controller placement is obtained by traversing all the available locations in each sub-network and the optimal number is depended on the density-based clustering. Then SDN is built by linking all switches to their neighboring controllers. Moreover, the size of each subnet was determined by the capacity of the deployed controller. Evaluations were performed on a set of network topologies in terms of computation time, fault tolerance, and propagation latency. Results obtained in comparison with other approaches showed the strategy achieved a fast response time, stable optimal solution, and can easily be applied to real-world networks.

In the same vein, Xiao *et al.* [55] proposed a network partitioning based CPP for an SDN-based WAN. The authors considered propagation latency, reliability, and load balancing as the important performance metrics and proposed a Spectral clustering Placement Algorithm that achieves the task of large network petitioning into different domains. The goal of the CPP approach was to achieve an optimum number of controllers' selection and their location in the WAN while maximizing the reliability and minimizing latency as well as balancing loads among controllers. The effectiveness of the algorithm was evaluated via simulation using Matlab based framework: Beacon controllers and Cbench as well as topologies from Topology Zoo: OS3E Internet2 topology. The results obtained showed the efficacy of the algorithm in solving SDN-based controller placement. The authors also advocated for the need to automate the determination of clusters for the placement in the future. Similarly, Xiao *et al.* [55] work is extended by Xiao *et al.* [56] to further address issues of network and domain partitioning in the SDN-based WAN. The authors exploited the benefits of spectral clustering to spectral clustering-based partition and placement algorithms known as a K self-adaptive SDN controller placement. The self-adaptive algorithm is based on matrix perturbation theory and employs eigenvectors to determine SDN domains' stability and automatically decides the optimal number of controllers [56]. That is, the approach effectively and efficiently partitions SDN-based WAN into multiple and optimal clusters while maximizing the reliability, minimizing latency. Moreover, the approach considerer processing latency, throughput, and reliability as the three performance metrics. The performance of the algorithm was also evaluated in the same way as in [55]. The results confirmed the effectiveness of the approach in terms of domain partitions, CPP and the future work asserted in [55]. The authors also stated the need to expand their future works to network latencies. In another related work, Cai *et al.* [59] CPP approach for an SDN-based WAN is also based on network partitioning into multiple domains. The goal was to guide against load imbalance among controllers or negative impacts on reliability while

TABLE 2. Linear and quadratic programming CPP strategies.

Ref.	Method	Implementation Tools	Evaluation Metrics	Network type
[13]	ILP	Simulations: C++, using the CPLEX 12.6 PC with 8 cores and 64 GB of RAM Topologies: Germany50 and CORONET CONUS	maximum SC delay and maximum CC delay	CN
[26]	IQP	Simulation: CPLEX Optimizer software. NTU Campus map	Communication latency, number of controllers, Workload distribution Neighboring RSU's location status	VANET
[4]	ILP	Simulation: GUROBI optimizer, Internet Topology Zoo (138 topologies)	Resilience, Number of controllers and Controllers utilization	Hybrid SDN/Legacy Networks
[36]	ILP	Simulations: 3 Topologies: Internet2 (10 nodes, 15 links), RNP (27 nodes, 33 links) and GEANT (40 nodes, 61 links)	Resilience, Reliability Load Balancing, path diversity	CN
[37]	Mixed ILP	Simulations, Gurobi solver, Topologies: Polska(12,18), germany50(50,88), Cost266(37,57), jano-us-ca(39,61), India35(35,80)	Resilience, Reliability	WAN
[40]	LP	Simulations: CPLEX optimizer 12.5, PC that has 2 Intel Xeon X5675 processors: 3.07 GHz, memory of 96 GB, Topologies: 10, 20, 30, 40, 50, 75, 100, 150, and 200 switches	Number of controllers, Cost of controllers (installation, lining and linking)	CN
[41]	LP	Simulations: CPLEX optimizer 12.5, PC that has 2 Intel Xeon X5675 processors running at 3.07 GHz with total memory of 96 GB, Topologies: 20, 25 and 30 switches	Number of controllers, Update Cost of controllers (installation, lining and linking)	WAN
[44]	ILP	Simulations: real ISP topologies	SC delays, CC delays Reaction time	WAN
[57]	MILP	Simulations: MATLAB, Mininet (ONOS), Real edge topologies (MANIAC and Ad hoc), CPLEX	Traffic Delays, Overheads	Edge

achieving minimum average delay between controllers and switches. To achieve this, the authors proposed a Load Balanced CPP which was expressed as a capacitated k-median problem and solved using an approach based on topological potential and minimum-cost flow [59]. The approach is three-phased: controller initialization for location determination based on topological potential, assignment of switches to controllers with minimum average delay based on minimum-cost flow, and placement update of controller locations and assignments [59]. The approach was evaluated via simulations using real WAN topologies and compared with a clustering-based network partitioning algorithm. The results obtained showed the approach effectively minimized SC delay, maintained good controller load balance, and reduce controller numbers to eliminate overload during controller placement.

Furthermore, due to the different drawbacks of the k-means algorithm, a clustering-based network partition algorithm was introduced by Wang *et al.* [47]. It was aimed at reducing the propagation latency due to the placement of multiple controllers in a WAN environment with queuing latency in controllers as an additional objective. The approach was evaluated using real-world network topologies and the results obtained showed improved performance by reducing the maximum latency.

B. LINEAR AND QUADRATIC PROGRAMMING BASED CPP

There several CPP strategies under linear programming (LP) and quadratic programming (QP) as summarized in TABLE 2 and are discussed as follows: Sallahi and St-Hilaire [40], [41] were the first to consider economic cost as one of the important metrics when finding the location of controllers. In [40], they focused only on the cost of controllers and proposed an LP model to solve the problem. The CPP scheme minimizes controller deployment cost while searching for the optimal number of controllers, location, types, and the links with different processing capacities between them as well as the bandwidths. Simulations results showed the scheme can only be used for small-scaled SDN and is not time and space-efficient as 10% of the problem could not be solved within the required time of 30 hours as well as running out of memory. Also, the scheme was designed for various enterprises as cloud-based networks. On the other hand, Sallahi and St-Hilaire [41] aimed to minimize update costs in the event of expanding or adding extra switches to the current network. LP was formulated utilizing the network design, list of extra switches to search how network reorganization that minimizes the update cost. Simulations were also performed and results showed its effectiveness in planning a new network or updating existing ones. Moreover, adding a controller

requires 5 links and the computational time is approximately 4.60 ms.

1) INTEGER LINEAR PROGRAMMING

The schemes that formulate the CPP based on integer LP (ILP) are discussed as follows. Zhang *et al.* [44] proposed a CPP scheme for a distributed SDN based on Pareto-optimal placement enhanced using a heuristic approach to improve SDN or SDWAN scalability and reliability. They proposed a low complexity or evolutionary algorithm to find the corresponding Pareto frontier in large networks. CPP was formulated as an ILP problem to search for optimal placement that minimizes the reaction time or delays apparently at the switches and also proposed an approximate algorithm is to solve it. Performance metrics considered were traffic exchanged between controllers and switches (SC-latency) and between controllers (CC latency). Moreover, evaluation using real internet service providers' network topologies showed an optimized controller choice acting as a data owner can improve reaction time by 2-4 times, and choosing a master controller of a switch as closest to the controller does not always minimize the reaction time obvious at the switches. Thus, the master controller of a switch selection must be tied to the optimal controllers' placement. Furthermore, a new quantitative tool was provided to optimize the planning and the design of the network supporting the control plane of SDNs for networks characterized by a very large and in-band control plane. Similarly, Muller *et al.* [36] proposed a CPP scheme called Survivor to enhance the survivability of the SDN control plane against failures. The approach employed adaptive heuristics that incorporate steps such as greedy procedure, local search, and evolutionary mechanisms. To ensure the SDN survivability, ILP was formulated which explicitly accounts for controller capacity, path diversity, and failover to ensure uninterrupted network connectivity using a backup list and prevent the controller from being overloaded using a capacity-awareness mechanism in the CPP strategy. Simulations on three real network topologies showed the superiority of the Survivor against other approaches. Path diversity significantly increases the survivability of the SDN, about 66% for single link failures, and having capacity-awareness in the solution can reduce the load of the overloaded controllers in both normal and failover states.

In a similar study, Das and Gurusanmy [4] proposed an efficient hybrid CPP (hCPP) approach for a hybrid SDN and legacy networks called hINCEPT- INcremental ContollEr PlacemenT. It was aimed at setting a migration path for network operators and to maximize switch to controller control channel resilience by minimizing the amount failure probability of SDN switch's control channel throughout the legacy network to SDN migration planning stages. It enhances effective decision making in deciding legacy nodes replacement with SDN switches, when, where, and how many controllers that can augment the resiliency [4]. An optimization problem was formulated over time using an ILP. The performance metrics considered were the number of controllers, channel resilience, and controller utilization. Moreover, a comprehensive evaluation performed against other approaches showed hCPP achieved a considerable higher resilience regardless of the number of controllers. About 77% higher resiliency was achieved with about 33% fewer controllers with 200% utilization as a consequence. Santos *et al.* [13] also proposed a robust CPP strategy to guard the network against the number of malicious attacks that target nodes intending to cause maximum disruption of network operations. The strategy utilized an ILP model to compute the feasible solutions given the minimum average delays of both switch-controller and inter-controller. It maximizes the reliability and availability of the control plane and the switches such that, in the event of controller nodes failing due to attacks, a switch path can still exist to other controllers not under attacks. Simulations were performed on two large network topologies using diverse malicious node attacks that are consistent with different strategies of an attacker. Furthermore, the robust solutions obtained were compared with non-robust solutions to determine if the minimum number of switches that can still be connected to at least a controller is maximized in terms of average SC and CC. The approach gained significant robustness for both sparser networks on an average basis and for more nodes shutting down due to attacks, achieving less SC and SC delays.

Furthermore, Vizarreta *et al.* [37] proposed a Reliable Controller Placement (RCP) to increase the availability of the control plane by protecting it against single link and node failures as well as providing continuous failover backup control paths through resilient routing principles. The strategy employed mixed ILP and emphasized on SC delay employed two control paths between switches and the controller assigned to it disjointedly or two replicas of the controller to each switch with disjoint control paths. The first case is called RCP- disjoint control path (RCP-DCP) while the latter is known as the disjoint the controller replicas (RCP-DCR). Both approaches were designed to enable fast and efficient failover with minimum effect. Simulations performed on real network topologies show significant improvement in the resilience of the control plane for both RCP-DCP and RCP-DCR but with additional cost to the average control path with about 2% increase in length for most topologies. Moreover, for link failures, both RCP-DCP and RCP-DCR produce the same performance with an applicable strategy determine by topological characteristics and controllers' number. While for node failures, RCP-DCP performed better as well as protecting controllers against failures. Lastly, Qin *et al.* [57] proposed the application of SDN to edge network architectures to enhance its control capability. By applying SDN, the control logic is taken off the data plane elements and onto the controllers, external entities. For the effective deployment of multi-controllers, a MILP was proposed to solve an optimization for CPP in a fog architecture specifically at the edge. Moreover, traffic delays and overheads where the considered performance metrics. The authors evaluated the effectiveness via proof-of-concept implementation using four devices and

TABLE 3. Bio-inspired CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[5]	APSO	Simulations: MATLAB 7.14 (R2012a EXATA 5.3,Intel (R) Core (TM) i5-7200U CPU at 2.50-GHz	Delay, delay jitter, link traffic, reliability, controller load, and economic Expense.	Satellite
[3]	SSOA with chaotic maps	Simulation: MATLAB, Intel Core i5 processor Internet Topology Zoo	time utilization, cost utility	CN
[34]	PSO	Simulations: MATLAB, MATLAB GA/GAmultiobj solver, ISP network with system 3561 from Rocketfuel	SC Delay, CC Delay, Controller load imbalance, No of controllers, Switch assignment	WAN
[2]	FFA	Simulations: MATLAB R2014a, Intel Core i5 with 4-Core processors	SC latency, CC latency, multi-path connectivity between the switch and controller	WAN
[6]	ABFO based MOCP scheme	Simulations: MATLAB 3 Topologies: Internet2 OS3E (34 nodes and 42 edges), Tinet (Internet Topology Zoo) (53 nodes and 52 edges) and RF-II (108 nodes and 306 edges)	reliability, load balance among controllers, and SC latency, CC latency – propagation latency	CN
[60]	VBO	Simulations: MATLAB, Python language Topology: Abilene, Savvis, Biznet, Internet2 OS3E	Latency, Load Balance (capacitated & incapacitated)	WAN

three wireless links. Others include the CPLEX tool and simulations performed on Matlab using ONOS controllers running on Mininet using MANIAC and ad hoc which are the two real edge network topologies. Accordingly, MILP was compared to a randomized greedy algorithm. The results obtained revealed the criticality of delay to controllers' location and overheads amount in terms of CC and SC. The authors also stressed the need to analyze extra mechanisms of creating a controller cluster, explore the impact of leader selection using RAFT in CPP, and so on in their future works.

2) INTEGER QUADRATIC PROGRAMMING

In a similar approach to [4], [13], [36], [37], [44], Liyanage *et al.* [26] also proposed a novel CPP scheme and a hierarchical distributed software-defined vehicular network (HD-SDVN) architecture for VANETs' roadside unit (RSU). The approach was based on the p-median facility location problem to determine the optimal number of controllers, their placement while minimizing latency communication overheads, etc. The approach divided the control plane into a top and bottom tiers where the top tier of the controllers was distributed on the Internet on a regional scale while several selected RSU were used as placements for the bottom tier to minimize latency. The optimization problem was formulated using IQP which is heuristic in nature. Several performance factors were considered such as the number of controllers to deploy, latency, the significance of RSU's geographical location, workload distribution, and vehicular statistics around the RSU location. Simulations results revealed the achievement of low latency by the proposed CPP strategy and the outperformance of both Internet-based SDVN and traditional VANETs by the HD-SDVN architecture in terms of low latency while preserving low communication overheads and high packet delivery rate.

C. EVOLUTIONARY BASED CPP

This section presents different CPP strategies that utilities the evolutionary computing approach to achieving the number, location, and allocation of controllers in the SDN environment. These are discussed as bio-inspired and Genetic Algorithms (GA).

1) BIO-INSPIRED CPP

There are several bio-inspired computing approaches to SDN-based CPP which focused more on creating a hybrid algorithm via optimization. This includes algorithms such as Particle Swarm Optimization (PSO) [34] Accelerate PSO (APSO) [5], and Salp Swarm Optimization Algorithm (SSOA) [3], Adaptive Bacterial Foraging Optimization (ABFO) [6], and Firefly algorithm (FFA) [2]. TABLE 3 shows the summary of each scheme.

Liao and Leung [34] introduced a CPP scheme to deal with both controllers' location and its switch assignments to conserve energy and controller migration. It was based on DCPP formulated as a multi-objective genetic algorithm (MOGA) with a PSO based mutation function. To search a very large search space, PSO based mutation keeps a pre-computed global best position of each objective and only uses the optimal position to generate velocity or guide the mutation parent in a very short time. Metrics considered include the SC delay, CC delay, and the controller load imbalance. Implementation evaluation showed PSO-based MOGA outperformed the original MOGA in terms of accuracy in shorter convergence time when optimizing a single objective that is much closer to real global optima. Moreover, optimizing multiple objectives also yields better accuracy in a shorter convergence time with a large search space.

Similarly, Wu *et al.* [5] proposed a CPP framework for software-defined satellite networking (SDSN) which is

considered from two perspectives: dynamic CPP (DCPP) and static CPP (SCPP) to achieve a comprehensive improvement in the performance of SDSN. While number and locations are adjusted based on cash sufficiency in DCPP, a limited number of controllers and the optimal location is found before the deployment in SCPP. In both cases, a multi-objective optimization model was formulated which uses heuristic algorithm APSO to solve it. Pareto Fonts were computed and with APSO, the local optimal solution obtained based on a gradient-based method when particle swarm achieved a certain degree of diversity. Moreover, several metrics considered such as delay, jitter delay, link traffic, reliability, controller load, and economic cost with further consideration of high dynamic attributes of the satellite network. Simulations result showed improvement in the SDSN performance as DCPP and SCPP outperformed the traditional SDSN in all the metrics used, with SCPP having less economic cost while DCPP has better flexibility to modify network settings. In the same vein, Ateya *et al.* [3] proposed a CPP solution strategy that is both latency aware and cost-aware for large scale SDN networks. It utilized a dynamic optimization algorithm to achieve an optimum number of controllers and the assignments of distributed switches to the available controllers. SSOA was developed using chaotic maps to improve performance by minimizing both latency and the cost of deployment. Metrics considered were cost, SC and CC latency, reliability, and computation time. Simulations were conducted and compared with other approaches such as the meta-heuristic and the game theory-based. Results show improvement in performance, given the reliability and the computation for the proposed strategy.

In another study, Zhang *et al.* [6] proposed a CPP scheme for the large-scale SDN to find the optimal controller placement, SC assignment, and the optimal number of routing requests to be handled by each deployed controller. A multi-objective optimization controller placement (MOCP) problem was formulated and proposed a bio-inspired and heuristic-ruled algorithm called ABFO to solve it. ABFO computation rules were redefined and the chemotaxis of the ABFO was customized. Performance metrics considered were reliability, low latency, and load balancing among controllers. In this case, reliability was maximized by minimaxing the control path failure probability, the load was balanced among controllers by minimizing the variance of controller load rate while low latency was achieved by minimizing the control path latency. Evaluations on three network topologies showed the effectiveness and improved performance of the proposed scheme when compared to PSA as well as its significant impact in network planning. The reliability probability of the heuristic was about 0.199% and 0.015% higher than PSA and ABFO respectively while the load balancing of ABFO was about 96.9% and 84.9% lower than the heuristic and PSA respectively. Moreover, the worst-case latency of ABFO was about 36.1% and 15.3% lower than the heuristic and PSA respectively. Similarly, Sahoo *et al.* [2] proposed a CPP strategy guard against single link failure at the reduced delay in communication to enhance the performance of the SDN. It considered CPP as a multi-objective combinatorial optimization problem and employs meta-heuristics approaches that are population-based to solve it such as PSO FFA. There were used to find the optimal number and placement of controllers, optimal switch-controller assignment, as well as the backup path for survivability. Three metrics such as controller-switch latency, controller-controller latency, and multipath connectivity between the switch and controller were considered. This was to ensure the design and development of a reliable control plane by way of optimizing the competing metrics which are latency and survivability. Simulations performed on different network topologies showed an improved reduction in the average delay in the event of single link failure. Also, FFA is considered time efficient in terms of computation and generated the optimum result than the particle swarm optimization.

In a similar work, Singh *et al.* [60] also proposed an efficient CPP approach for SDN-based WAN that is heuristic in nature and decreases the total average latency of the SDN network between switches and controllers as well as between controllers to maximize SDN performance. The authors developed a new optimization algorithm known as varna-based optimization (VBO) to solve CPP where switches and controllers are represented as particles. VBO is considered compared to other optimization-based solutions since the same formulation is not used for population particles, particles in a given particle class in one generation do not necessarily remain in it, and Varna class is not dictated by birth but by particles' fitness value called Karma [59]. The approach considered capacitated, incapacitated, load-aware capacitated, load-aware incapacitated, and latency. The effectiveness of the approach was evaluated via simulations on Matlab with real-world topologies and results compared with other approaches such as PSO, Jaya algorithms, etc. The results obtained showed the superiority of the VBO solution over other CPP solution techniques.

2) GENETIC ALGORITHM BASED CPP

The CPP schemes based on GA as part of the evolutionary-based techniques are summarized in TABLE 4 and discussed as follows. Hu *et al.* [42] CPP scheme minimize the energy consumption of the network's control traffic under the constraint of the delay of control paths and the controller loads. It employed an optimization model called binary integer program (BIP) for small-scaled networks and a genetic heuristic algorithm called genetic controller placement algorithm (IGCPA) was designed to search an effective sub-optimal solution in large scale networks. Simulations performed on several network topologies show improvements in energy consumption. It showed the heuristic algorithm was closed to the BIP solution in terms of energy-saving and for links with the same energy consumption, no more than 4% extra links were used by the IGCPA. Also, Jalili *et al.* [9] proposed a Switch to Controller Assignment Process to improve QoS in the SDN. It employed an AHP algorithm for multi-criteria

TABLE 4. Genetic algorithm based CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[9]	Genetic Algorithm hybridized by AHP technique	Simulations: MATLAB 2016a 4-GH Intel Core i7 machine, Internet topology Zoo	propagation delay, hop count and link utilization	CN
[14]	Kuhn-Munkres algorithm Genetic algorithm	Simulations: Intel Core i7 (2.9 GHz) Topologies: TA2, GERMANY50 and INDIA35	Propagation delay and controllers load balancing	WAN
[12]	MHNSGA	Simulations: MATLAB, 2014a, 4-GH Intel Core i7 Internet topology Zoo	SC latency, CC latency load balancing	WAN
[30]	MHNSGA-II	Simulations: MATLAB, Intel Core i7 CPU, 4 GH. Internet Topology Zoo	SC latency, CC latency, load balancing	WAN
[42]	Genetic Algorithm (BIP)	Simulations: MATLAB, IBM ILOG CPLEX Optimizer, 2 Intel Xeon, E5-2430 processors with 8 cores, equipped with 16 GB of RAM, 4 topologies: Abilene (12 nodes, 15 links), Janos-us (26 nodes, 42 links), Pioro (40 nodes, 89 links), Zib (54 nodes, 81 links)	Energy Consumption, control paths delay, controller load	CN
[51]	CGA-CC, GD-based scheduling algorithm, greedy algorithm	Simulations, Topology: Sprint network (Asia Sprint network: 14 nodes with 23 links Europe Sprint network: 15 nodes, 22 links Global Sprint network: 82 nodes, 1056 links)	Propagation latency, load balancing	WAN

decision and weight preference selection, optimal assignment of switches to controllers as well as the control path for each pair. Moreover, they proposed a technique called controller placement GA to solve the CPP problem of the optimal number and locations of controllers in the network. Metrics considered were controller load, latency, hop-count, and link utilization. Simulation results showed efficiency in the CPP and optimal assignment as well as the outperformance of latency-based assignment by multi-criteria assignment in terms of load balancing.

In [12], Ahmadi and Khorramizade proposed a large-scale multi-objective controller placement approach using a fast and efficient adaptation of evolutionary algorithms. They utilized a heuristic algorithm known as Multi-Start Hybrid Non-Dominated Sorting Genetic Algorithm (MHNSGA) while considering performance metrics such as SC-latency, CC-latency, load balancing, and reliability for link failures in the objective function. It requires sound memory resources and is targeted at computing the Pareto Optimal Control Placement (POCO) [23] using the Pareto front in the objective space and the Pareto set in the decision space. The algorithm is greedy, producing a high-quality initial population and a fast Pareto finder. The evaluation was performed on several network topologies to assess MHNSGA performance in comparison with other approaches such as POCO, Pareto Simulated Annealing (PSA), and PSO-CGLCPP in terms of time and search space. Results showed MHNSGA outperformed others given less computation time and space, having an average deviation of about 0.8% in comparison with the original Pareto optimal set. The scheme can explore a large portion of the search space and generate with a high degree of accuracy, estimation of the Pareto optimal front which was about 20 times faster. Moreover, MHNSGA proved its efficiency and superiority in solving MOCCPP over other efficient algorithms. In a similar study, MHNSGA in [12] was improved by Jalili *et al.* [30] using a heuristic CPP strategy called Multi-Start Hybrid NSGA-II (MHNSGA-II) for large-scale networks to achieve efficient resource utilization. MHNSGA-II, an adaptation of the NSGA-II algorithm is formulated as a multi-objective optimization model considering performance metrics of latency between nodes and assigned controllers, latency among controllers, and load balancing in its objective function. In this strategy, Pareto optimal solutions are obtained by computing different and accurate estimation of the Pareto optimal set. Its mechanisms include greedy initialization, fast Pareto finder process, local search, dispersion, and multi-start strategy. Evaluations performed against the POCO framework [23] showed efficiency (i.e. computation time and memory) in computing accurately different Pareto Optimal placements estimation set for any given metric. POCO was found to exceed the machine's RAM.

Furthermore, Yuan *et al.* [14] proposed a CPP scheme considered to be efficient and accurate for both placement and assignment problem in the WAN. It finds optimal solutions with minimum average propagation delay between switches and controllers. The scheme employed two algorithms: Genetic and Kuhn-Munkres to efficiently solve the controller placement problem. CPP was viewed as the challenge or the task of finding minimum weight matching of a bipartite graph where edges represent the link and nodes represent controllers/switches (i.e. distance between nodes). The Kuhn-Munkres algorithm finds minimum delay, the optimal assignment between the switches and controllers while the genetic algorithm solved the corresponding CPP based on the optimal controller assignment strategy. Propagation delay and load balancing were the performance metrics considered. Evaluations performed on three network topologies based on simulations showed the effectiveness and improved performance in terms of propagation delay reduction and improved load balancing among controllers.

TABLE 5. Heuristic and Greedy algorithm based CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[1]	Heuristic Algorithm	Simulations: MATLAB, 2018a and CPLEX 12.6, 4-GH Intel Core i7 machine Internet Topology Zoo (160 topologies)	SC latency, CC latency, and a new load balancing	CN
[7]	FTCP based Heuristic algorithm.	Simulations: Linux, Intel Xeon CPU at 2.67 GHz, Mininet and 124 Internet Topology Zoo	Number of controllers, Reliability, Load balancing	WAN
[32] [33]	Greedy Algorithm	Simulations: Internet2 OS3E, Rocket fuel topologies	Number of controllers Reliability	CN

Recent studies have shown that CPP cannot be solved in isolation and any effective CPP solution may heavily depend on the solutions to other problems such as resource or request scheduling, load balancing, security, etc. This was achieved in [43], while in [51], Huang *et.al.* proposed a framework for controller placement and scheduling problem (CPSP) that addresses the two problems collectively. The approach is a constrained optimization problem that enhances the control plane usage while ensuring low network response time. The controller scheduling problem (CPS) was solved effectively using a gradient-descent-based (GD) scheduling algorithm to optimize request distribution probability over all controllers to balance the scheduling performance and problem scalability trade-off [51]. In the same vein, the CPP was addressed using a Clustering-based GA with Cooperative Clusters (CGA-CC). It operates by partitioning the network into sub-networks devoid of overlaps to effectively place controllers in each and also to minimize the GA search space. Also, the greedy algorithm was employed to divest unexpected requests to neighboring sub-networks. Evaluations through a series of simulations using Sprint topologies showed the effectiveness of the two algorithms when compared with others. GD-based scheduling algorithm showed a 37.5% reduction in response time with high control plane throughput. Moreover, the CGA-CC response time and control plane usage performance is pointedly better than algorithms like the K-center and Multi-controller Selection and Placement Algorithm. Also, CGA-CC outperformed GA in large-scale networks in terms of low computation cost, low response time, and high control plane utilization.

D. HEURISTIC AND GREEDY ALGORITHM BASED CPP

Several CPP schemes are based on a heuristic approach to determine optimal controller location and allocation to optimize the reliability of the network as summarized in TABLE 5. Accordingly, Ros and Ruiz [7] proposed a CPP strategy by optimizing reliability in the placement, taking into account the failure probability of each component (node and link) within the southbound interface. This was specifically to solve issues of controllers' synchronization and the incessant consumption of switch resources by the controller. They employed heuristics as a solution strategy to minimize deployment cost and the Fault-Tolerant Controller Placement problem (FTCP) was formulated thereof where the operational path between multiple controllers and a switch is guaranteed by a given probabilistic value. The heuristic algorithm finds the optimal placement that meets reliability and an optimal number of controllers. Evaluation on a wide range of network topologies showed the number of controllers to deploy was positively related to the number of with one link in the network. For reliability, each node was expected to connect 2 controllers and 75% of topology, at most 8 controllers were adequate. According to the authors, the scheme is important to assist SDN operators to ease the concerns of installing a logically centralized controller in their networks and to promote research.

Similarly, Ateya *et al.* [3] proposed a CPP strategy focused on improving network scalability while achieving low latency among controllers in the network. Accordingly, a location-allocation model was formulated based on factors such as costs, types, and capacities of controllers to (1) find the optimal number of controllers deployed while minimizing cost, and (2) balancing the loads among different controllers while minimizing latency between controllers. Moreover, a heterogeneous cost placement model was designed to determine the optimal number of controllers to be deployed while using a heuristic approach to solve it. For load balancing among different distributed controllers, a capacity and load-aware SDN controller placement was introduced which utilizes controller placement anytime Pareto local search. Performance metrics such as switch-controller latency, inter-controller latency, and load balancing were considered in the optimization model. The models and algorithms were validated using simulations and the results obtained confirmed the importance of the parameters in terms of costs, fair load distribution in terms of the homogenous and heterogamous controllers, and efficiency. Hu *et al.* [32] also proposed a CPP scheme to maximize the reliability of the control network. It utilized three different placement algorithms and performed simulations on real network topologies to evaluate them. The algorithms are the l-w-greedy, simulated annealing, and the brute force. Results showed significant improvement in network reliability with promising latencies. Brute-force produced the optimal solution but consumed lots of time while annealing was closed to optimal. Thus, reliability was reduced when placing too many or few controllers. Accordingly, Hu *et al.* [33] also considered reliability by using a metric called the expected percentage of control path loss. A mathematical model known

TABLE 6. Simulated annealing based CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[29]	Simulated Annealing	Simulations: MATLAB -CPLEX optimizer 12.6.2, 2 Intel Xeon E5-2630 processors (2.4 GHz). Topologies: Internet Topology Zoo and Internet 2 OS3E.	SC latency with and without link failure	WAN
[31]	Pareto Simulated Annealing	Simulations: MATLAB, Internet Topology Zoo	SC latency, CC latency, resilience against node and link failures, load balancing	WAN
[35]	Simulated Annealing	Simulations, Real network topologies	Bandwidth utilization, Reliability	WAN

as reliability-aware CPP was formulated and explored several possible heuristic placement algorithms to solve it. Simulations performed using real topologies, found the number of controllers and their locations affects reliability significantly. They concluded that strategic placement can improve reliability with promising SC latencies. They confirmed that CPP is an NP-hard problem.

E. SIMULATED ANNEALING ALGORITHM BASED CPP

Simulated annealing has probabilistic characteristics that assist in terms of approximation and several CPP strategies have been developed based on it. These approaches are summarized in TABLE 6 and discussed as follows: Killi and Rao [29] proposed a capacitated controller placement (CCPP) strategy to guard against single link failure or its recovery with no significant impact on the worst-case SC latency. It employed simulated annealing and was designed to find the optimal number of controllers to be deployed that can minimize worst-case SC latency in the event of a single link failure while preserving the capacity of each controller. The approach known as Link Failure Aware Capacitated Controller Placement (LFACCP) was formulated as an optimization model with the consideration of two objective functions: worst-case latency and link failure. Two variants of LFACCP were also proposed which considers worst-case latency with and without link failure. Evaluations performed via simulations on different network topologies show that LFACCP can significantly lower the overall worst-case SC latency during a single link failure but with an insignificant rise in worst-case SC latency without link failures. Moreover, it significantly lowered the maximum and average CC latency for both with and without failures.

In another study, Shaoteng *et al*. [35] proposed a flexible deployment of a distributed control plane called a black-box optimization framework to implement simulated annealing for the association. It operates automatically to select the number of controllers, their locations, and the control region. Moreover, it also quantifies the impact of the consequent control traffic during the deployment by searching for where there is no congestion to meet reliability and bandwidth utilization requirements by the control traffic. Evaluation using real network topologies showed the approach close to optimal solutions under changing conditions. It also showed excessive bandwidth utilization and congestion when executing the CPP strategy without regarding the control traffic, with worst cases ranging between 20.1 %, to 50.1% and more. According to the authors, the scheme can be used as a tool for measuring and predicting the tradeoff between bandwidth and reliability by network service providers as well as operators.

In considering multiple objectives using a probabilistic based algorithm like simulated annealing, an improvement to the solution is needed. Lange *et al.* [31] proposed a CPP strategy based on the POCO framework [23] which works well with multiple-objectives. It employed PSA to search all candidate solutions of Pareto-optimal placements which returns the Pareto frontier of the subsets to achieve less computation time in large-scale networks with less accuracy. PSA explores search space by moving to the neighbor placement with a slow decrease in the probability of adding worse placement to load search space. It focused on more than one failure scenario for nodes and links but did not take into account the operational probability of all nodes, links, and controllers in the problem formulation. Factors considered are the SC and CC latencies, balancing, and resilience against both node and link failures. Simulations were performed on real networks and evaluated in terms of time and accuracy, which was 20 times faster. Also, the POCO tool was developed.

F. OTHER CPP STRATEGIES

Other CPP strategies have been proposed and developed. These include the cooperative game theory [20], the Clique-based [39], Resilient mapping [10], Switch migration based [27] and the ML-based [16], [53]. The summary for each approach is presented in TABLE 7.

1) COOPERATIVE GAME-BASED CPP

In this CPP scheme, Killi *et al.* [28] proposed an SDN-based CPP strategy with the goal of latency minimization. It partitions the network into sub-networks using the k-means algorithm like in [43] with cooperative game theory initialization such that each partition has at least one controller. That is, it modeled the network partitioning as a cooperative game where a set of all switches were the players and the switches maximize their value by forming alliances

TABLE 7. Other CPP strategies.

Ref.	Method	Implementation	Evaluation Metrics	Network type
[10]	Resilient Multi-controllers Mapping	Simulations: MATLAB IBM CPLEX optimizer. Small scale AT&T network, medium-scale GEANT network of Internet Topology Zoo	Propagation latency, resilience, cost, Back-up capacity	CN
[27]	Switch Migration Based Controller Placement (SMBCP) algorithm	Simulation; Python. CONUS; topology of 60 nodes	Load balancing, cost: cost of the migration request and load change cost	Optical
[28]	Cooperative k-means/ Cooperative game	Simulations: Python, Internet 2 OS3E topology and Internet Topology Zoo	Worst-case latency, controller load	WAN
[39]	Clique-based scheme	Simulations: Internet Topology Zoo (43 nodes)	SC latency, CC latency, Controller capacity	WAN
[50]	Bargaining game theory	Simulations: IBM CPLEX, MATLAB, CVX 2.0	Load balancing, Latency, switch-controller and controller-controller communication overhead	WAN
[48]	Matching Theory and Coalitional Games	Simulations: MATLAB Topologies: Fat-tree and VL2 Real-world datacenter	Cost, Load balancing	Data Center
[49]	Non-zero-sum based game theory	Simulations: MATLAB Random network of 28 Switches, 7 Controllers	Cost, packet drops and delay	WAN
[16]	ML	Simulations: Python on Ubuntu Server, 14.04.1 LTS with 16 CPUs 6 topologies: Topology Zoo (15), AttMpls (25), Bics (33), Cernet (40), Uninett2010 (74), Deltacom (99) and Cogentco (180)	traffic load weighted average control latency	CN
[53]	ML-based heuristic algorithm	Simulations, C# Language Topology Zoo: (Abilene-11nodes, 14links), (InternetMCI-19nodes, 45links), (Geant2010-37nodes, 58links) and (Iris – 51nodes, 61links)	Propagation latency	WAN

with other switches. Moreover, two load-aware cooperative k-means strategies were proposed to circumvent partition imbalance concerning the size. Evaluation via simulations showed the proposed strategy based on random initialization produced solutions closed to optimal in terms of worst-case SC latency. In comparison, it also outperformed the standard k-means algorithm, and efficient with less than a second for topology having 109 nodes. Moreover, one of the two variants load-aware cooperative k-means strategies generated balanced partitions with few partitions while the other always produce balance partition regardless of the number of partitions.

In [48]–[50], the authors also proposed game theory-based algorithms to address CPP in the SDN. However, the evaluation of the algorithms is based on different network types including data centers [48]. Ksentini *et al.* [50] used the bargaining game theory to find the tradeoffs between multiple objectives including latency and communication overhead between switches and controllers, between controllers and also the load balancing between controllers. Evaluation via simulations and geometric programming shows the effectiveness of the proposed approach in finding the optimal controller placements. Also, Wang *et al.* [48] presented a datacenter approach to address the issue of long response time and high cost of maintenance to traffic dynamics caused by the static assignment of switches to controllers. Accordingly, a dynamic controller assignment problem (DCAP) was formulated as an online optimization to minimize the issues. DCAP was decomposed using a randomized fixed horizon control framework into chains of stable matching problems and to achieve a small loss in competitive ratio. To efficiently solve the CPP, a two-phase algorithm was proposed that incorporates important concepts from matching theory and coalitional games. Evaluation via theoretical analysis shows a 46% reduction in the total cost and a better load balancing. Similarly, Rath *et al.* [49] proposed a non-zero-sum game based distributed technique which is simple and real-time based and less complex processing. The technique to address CPP operates by using an optimization engine at each controller to calculate a payoff function, compared with its neighbors, and make informed decisions. The decisions that can be taken include adding new controllers, delete existing controllers, and performing dynamic offload between controllers. The approach was evaluated through simulations to verify its usability and proposed a deployment framework that can improve QoS and save significant cost in an OpenFlow enabled setting.

2) CLIQUE-BASED CPP

In this category, Tanhan *et al.* [39] proposed a Resilient Capacitated CPP (RCCPP) to find the optimal number of controllers, placement, and assignments to switches.

Two algorithms were proposed where the clique-based approach in graph theory, having a polynomial time complexity used for finding the optimal and cost-saving solutions heuristically. To search the maximum cliques, it narrows down the search space and the optimal solution was found as a subset of one of the maximum cliques. To meet the traffic load of switches, both the SC and CC latency requirements and the controller capacity were considered. Simulations performed on real network topologies showed average utilization of the controller was between 80% to 90% against single or two controller failures. According to the authors, the approach can be used service providers as a guide for resilient SDWAN design.

3) RESILIENT MULTI-CONTROLLERS MAPPING CPP

In this perspective, Kili and Rao [10] CPP scheme was designed to achieve optimal controller-switch assignment with minimum cost, the controller capacity, and average and worst-case switch-controller latency while ensuring full resiliency of the control plane. The approach deployed three optimization models such as resilient multi-controller mapping with minimum cost (RMM-MC) where each switch sends a fraction of request to each of the controllers allocated to it, resilient multi-controller mapping with minimum backup capacity (RMM-MB), and resilient multi-controller mapping with latency minimization (RMM-LM). The goal was to guard against a pre-specified number of controller failures. Evaluation on two networks showed the proposed models achieved minimum controller's number with at least 50% reserved backup capacity minimization with a higher resilience against controller failures and a lower variation coefficient in terms of switch-controller latency in comparison with other approaches.

4) MACHINE LEARNING BASED CPP

In another CPP scheme, ML techniques were employed. He *et al.* [16] proposed a scheme based on ML for SDN where a multi-label problem was formulated to investigate different algorithms that learn from the distribution of traffic using feature vector to predict global network placements. The goal was to maximize the speed of existing heuristic approaches and predict accurate initial solutions for local search algorithms. It exploited changes in traffic patterns with time and applied a heuristic algorithm to generate several solutions used for training the ML module. With the trained dataset, the ML module processes the new traffic patterns which then produce initial solutions that were improved by the heuristic algorithm module. A case study was performed through simulations using a K-median problem for optimization using ML algorithms such as neural networks (NN), logistic regression, and decision tree. The results showed NN produced the best abstraction as an initial solution for the heuristic approach and could save a significant amount of the algorithm time. That is, for heuristic algorithms with an initial feasible solution, incorporating ML for the prediction can yield high-quality solutions with reduced algorithm runtime. Similarly, Mostafaei *et al.* [53] proposed a CPP scheme for the SDN networks using an ML-based simple heuristic algorithm. The scheme was aimed at minimizing the overall propagation latency among the various controllers and switches. In other words, the considered a single objective in the optimization problem and used it to identified optimal placement. To achieve this, each node was trained with learning automation which determines the nodes' role and used in the selection of controllers to reduce the propagation latency which is set as the input to the algorithm. The network was then split into sub-networks based on predefined threshold values by the learning algorithm controller placement (LACP). Performance evaluation via simulations using Topology Zoo showed LACP requires less propagation latency than other approaches such as k-means, etc. when placing multi-controllers in the SDN environment.

5) SWITCH MIGRATION BASED CPP

In this category, Zhao *et al.* [27] CPP strategy for the SDN-based optical network (SDON) focused only on the load balancing optimization among distributed controllers. They employed a multi-controller coordinated deployment strategy called SMBCP which adopts the hierarchical architecture controller which is based on parent-child collaboration mechanism or switch migration. It operates by first computing the load and load ratio matrix of each single domain controller to determine the triggering factor (TF) which is then traversed and assessed against a given threshold. If the triggering factor $>$ threshold, it is added to set TF otherwise the algorithm terminates. Also, the triggering factor was traversed in TF and then added the controller to the set C_{EM} which is the overload controller set. Finally, switches with maximum migration ability were taken and controllers not controlling the switches counted to compute the migration efficiency by selecting to get the maximum value. The last steps executed repeatedly till traversal was completed and the algorithm halt. To evaluate the strategy, SMBCP was assessed against two algorithms: closest switch migration algorithm and maximum utilization switch migration algorithm which aim at minimizing energy consumption in traffic control and controller load balancing. Results showed the SMBCP algorithm considerably improved load balancing as well as minimize switch migration cost.

V. DISCUSSIONS AND POSSIBLE RESEARCH DIRECTIONS

SDN model addresses the inherent challenges faced in the traditional networks via network programmability which ensures flexibility, efficiency, and innovations in the network [3]. However, the networking paradigm is faced with challenges in terms of multiple or distributed controller's deployment. Challenges such as scalability, consistency, reliability, and interoperability [19] were identified as parts of the key challenges confronting the design of an efficient and robust distributed SDN controller platforms. Moreover, from the perspective of the SDN scalability, many other

challenges affect its realization such as controller placement, controllers' failure, flow setup latency, consistency [18]. Among these challenges, controller placement constituted a critical problem to the realization of SDN scalability while achieving other important network requirements. This problem is termed the CPP and several approaches have been proposed or developed in this regard for large-scale SDN networks.

This paper focused on CPP and we performed an in-depth review and analysis of some of the existing CPP strategies involving multiple controller deployments for the management of the SDN. Several studies were considered and several strategies were identified which are either cluster-based [8], [17], [38], [43], [54], [58], [59], LP/QP [4], [13], [26], [36], [37], [40], [41], [44], [57], bio-inspired [2], [3], [5], [6], [34], [60], GA [9], [12], [14], [30], [42], heuristic-based and greedy algorithm [1], [7], [32], [33], simulated annealing [29], [31], [33] and others [10], [16], [27], [28], [39]. The summary of the strategies in terms of the algorithm used, implementation, performance metrics, and network type were captured in TABLE 1-7. The analysis revealed that CPP is a critical problem that can be achieved with careful planning since several conflicting factors influence the SDN performance such as reliability, controller load balancing, latency, operational costs, etc. as shown in Fig.2. The CPP algorithm is dominated by the network portioning or clustering approaches, followed by the LP mathematical model-based, the heuristic-based strategies as well as the bio-inspired strategies. Moreover, the performance metrics in each strategy are optimized by either minimization or maximization to achieve the desire network performance and QoS. Several CPP strategies have considered multi-objectives in their optimization model while few considered single objectives such as cost, latency, reliability, etc.

Furthermore, all the studies considered in this review, designed their CPP for large-scale network except Hu *et al.* [42], which is designed for small-scale network. The analysis also revealed that each strategy was designed for a specific network type. For instance, the majority were designed for SDN-based WAN or CN, and only a single study considered other network types such as Satellite [5], Optical [27], WSN (SDWSN) [43], VANET [26] while [4] was designed for a legacy to SDN migration. Accordingly, implementations were performed using simulations either on MATLAB or real network topologies such as Internet topology zoo, Internet 2 OS3E topology, Deltacom, Cogentco, AttMpls, Bics, Cernet, Uninett2010, Rocket Fuel, and so on as shown in TABLE 1-7. It shows the various type of experimental tools that new researchers can utilize for solving different objectives, single or multiple objectives, and/or mainly CPP in the various network types.

A. POSSIBLE RESEARCH DIRECTIONS

Based on the review and analysis performed, several possible research opportunities were identified. There are discussed as follows:

1) EFFICIENT TRAFFIC ENGINEERING, LOAD BALANCING, ENERGY-AWARE AND COST-AWARE ALGORITHMS

In Lu *et al.* [15] and Wang *et al.* [22], there is a need for the design of an efficient CPP algorithm and a cost-aware algorithm. While Lu *et al.* [15] demanded online search algorithms that are time efficient, [22] advocated for efficient algorithms that consider QoS as a priority. For cost-aware, [15], [22], [46] calls for a thorough investigation of the trade-offs between turning on and off links or controllers to conserve energy and the resultant cost incurred. Other aspects that require further research include effective ways to configure virtual control charts and extra virtualization support functions for a full distributed virtual control plane [15] as well as attack-aware CPP involving combing actual network and attack mode to guide real-world controller deployment [13], [15]. Though [13] considered security in their CPP scheme, it is important to note that CPP cannot be achieved in isolation but collectively. Thus, attack-aware CPP should be designed to guard against malicious attacks while achieving important network performance requirements. Moreover, *Wang et al*. [22] further stressed that, due to the complexity and challenges of modeling multi-objective optimization, all the objectives should be considered since one or two are inadequate for the CPP solution. In the same vein, the benefits of multiple controllers in large-scale SDN-based networks can only be fully achieved if there are proper coordination and communication among controllers for good QoS. Zhang *et al.* [20] highlighted the need to properly coordinate between multiple controllers and switches in terms of different policies in large-scale networks with multiple domains to avoid conflicting issues. Furthermore, dynamic load balancing when deploying multiple controllers in large-scale networks has to be considered. In this case, traffic engineering should be an important consideration when formulating CPP and also to incorporate network virtualization and SDN with multiple controllers to bring about innovations.

Ros and Ruiz [7] also emphasized on the need for effective controller reassignment strategy for DCPP in SDSN, controller cooperation with each other, effective topological design of satellite constellation concerning controller placement, coverage rate, antenna pointing high precision attitude control support, energy-aware and attitude control routing protocol design in the SDSN for performance enhancement. Liao *et al*. [8] added that to discover the different strategies for CPP with routing and assignment as well as without routing and assignment, the impact of different metrics on the switch-controller assignment needs to be investigated. In the same vein, the works in [9] and [13] should be extended with metrics such as load balancing and energy-aware as objectives while that of [10] and [14] be extended with energy-aware in their objective functions for improving QoS in the SDWSN while still achieving resiliency in a pre-specified number of controller failures in [14]. Similarly, Zhao *et al.* [10] and Killi *et al.* [11] can be also extended with other performance metrics in a multi-objective manner.

2) RELIABILITY AND RESILIENCE

In the perspective of network reliability, Hu *et al.* [33] advised on different design choices such as configuring primary control paths as well as backup paths, assigning switches to some backup controllers besides their primary controllers. Similarly, Singh and Srivastava [23], emphasized the need to consider load balancing and all possible failures scenarios for all network partitioning based CPP. Besides, dynamic clustering should be incorporated into the CPP of large-scale networks to address controller utilization and random traffic patterns. Other open research opportunities that require attention include implementation and deployment of large-scale SDN, integrating QoS in function placement problem for NFV, finding the advantages and disadvantages of clustering mechanisms, ILP, greedy, heuristics approaches of CPP as well as efficient solution for a large-scale network with dynamic traffic load and fault [23]. Ros and Ruiz [12] also expressed the need to further investigate the cost of implementing high reliability in topologies with few nodes and degrees as well as why add extra links are much costlier than deploying a new controller in WAN and how the cost can be minimized. It is advisable that the proposed solutions in Tanha *et al.* [39] can be extended to handle node or link failures and with different objectives such as minimizing the expected control path, etc. Also, designing a dynamic RCCPP in which controller-switch assignments are changed based on time-varying traffic loads of the participating switches is important.

3) GENERAL AND MACHINE LEARNING BASED CPP STRATEGIES

To further improve the CPP scheme, the framework proposed by He *et al.* [13] needs to be tried with other ML algorithms for prediction as well as employ additional features, such as network connectivity, energy-aware, etc. in the objective function. More study is required to investigate the claim in [16] that ML algorithms can only be used as an initial solution for heuristic CPP approaches. Das *et al.* [45] suggested that approaches based on ML should be explored in dynamic environments for speed and efficient CPP solution while [46] advocated for traffic Internet model using suitable learning models by collecting real traffic in the SDN for traffic prediction based CPP. In Yuan *et al.* [14], the method for determining the optimal number of controllers in multiple domains network should be further investigated to ensure the costs of network construction and operation are under optimal control for scalability and controller load balancing. The dynamic design should also be adopted to minimize the response time or latency of the control plane. In Lu *et al.* [15], we considered it important to further investigate the implementation of the DBCP approach in a variety of applications and the ease of its extension given available parameters in the distance functions such as delay or physical distance as well as the placement objective function. Additionally, the impact of the DBCP approach in the maintenance stage of SDN concerning cost minimization is also open to further research.

Furthermore, the CPP scheme proposed in He *et al.* [16] should be further investigated to validate the claim that the proposed method outperformed other existing Internet-based SDVN and traditional VANETs in terms of latency, communication overheads, packet delivery rate, and other important parameters. Similarly, Das and Gurusamy [17] CPP scheme should be further investigated with additional constraints such as propagation latency, network traffic, controller load balancing, cost optimization, energy efficiency which can be extended to SDWSN by taking incremental controller addition into account. In Karakus and Durresi [18], a generic link failure-aware model that can account for a security measure in the CPP should be designed while Bannour *et al.* [19] advocated for fair exploration of the trade-offs among several metrics considered such as SC latency, CC latency, load balancing, etc. since they conflict with each other. Moreover, given the results obtained in [30], there is a need to extend the strategy to large-scale networks partitioned into clusters for the accurate, precise, and efficient result while the scheme in [53] should be extended by considering the number of controllers as input as well as exploring some of the ML algorithms.

4) NETWORK TOPOLOGIES AND SCALE-UP

In this point of view, Liao and Leung [34] CPP scheme results showed that global best positions were achieved in MOGA using heuristics. However, there is a need to perform evaluations on various network topologies and scales to confirm its effectiveness. In the same vein, the proposed scheme in Muller *et al.* [36] should be extended by considering other topologies and failure scenarios as well as exploring more aspects to enhance survivability in their model. Vizzaretta *et al.* [37] advised on efficient heuristics for multi-objective resilient controller placement. This is due to their approach being inept for online usage despite being fast while Liu *et al.* [38] advocated for a multi-objective to be considered in their optimization model to assess its effectiveness compared to a single metric while, further research is required for strategy in [1]. This is to confirm the performance of heterogeneous controllers against homogenous controllers in terms of the cost and load balancing among controllers given the different topologies' geometry in [1]. Lastly, the CPP scheme proposed in [54] should further be investigated to the possibility of defining the construction of tree topology from a given controller with the consideration of many network performance metrics since the SC shortest path is not the optimum approach to improving load and robustness of the control layer. Also, a load migration mechanism between controllers should be defined [54].

5) COST ANALYSIS AND REAL-WORLD TESTBEDS

In Sallahi and St-Hilaire [41], it is advisable to carry out an analysis of the impact of cost multiplier to remove existing network infrastructure and the effect of the quality of such

infrastructure on the output of the expansion model as well as the application of heuristic algorithms in solving their CCP. Accordingly, Kobo *et al.* [43] expressed that though SDWSN is still in its development stages and the lack of important support tools for simulation poses a serious limitation. Thus, efficient tools that support real-world testbed need to be designed and developed for SDN-based sensor nodes. In the same vein, the performance of the GD-based scheduling algorithm and CGA-CC algorithm has to be evaluated on a real-world testbed to validate the effectiveness of the approach in [51]. This is important to demonstrate that CPP cannot be achieved in isolation but collectively since it may depend heavily on other related problems. Rath *et al.* [49] also called for the need to extend the non-zero-sum based game theoretic CPP strategy to real-world SDN by developing an emulator platform with controllers and switches that are OpenFlow enabled as well as incorporating inter-controller communication technique that ensures proper assessment. Also, the scheme can be extended to the cloud network.

6) CPP APPLICATION AND MOBILITY-AWARE

Despite the existence of vast areas of SDN, CPP insights can also be extended to non-SDN contexts like emerging areas such as cloudlet placement and operator placement in the perspective of edge computing and networking [45], [46] for IoT, sensor networks, and so on. Kumari and Sairam [46] suggested it can be effective by exploring their unique topology, range, energy limitations, and other factors. For SDN-based performance due to user mobility, [46] has called for the use of mobility-aware controllers' placement which has been considered an open research issue.

7) CONTROLLER AND FLOW RELOCATION

The relocation of the controller can have a positive impact on the flow setup time. Accordingly, Kumari and Sairam [46] suggested that, instead of inter-domain controller relocation to improve load balancing, a centralized approach should be adopted where controllers' capacity is aware by an oracle that provides signaling. Also, [46] also suggested two issues that need to be taken into consideration when distributing a fraction of flow to other controllers instead of switch migration. They include maintaining a track of the flows and determine the correct flow division as well as simultaneously creating multiple masters of a switch when implementing flow relocation.

8) SWITCH MIGRATION, SWITCH-CONTROLLER SELECTION, MULTI CONTROLLER MAPPING PER SWITCH

In these perspectives, Kumari and Sairam [46] also stated that switch reassignment in a dynamic network environment comes with great overheads and expressed the need for more research to be done in this direction. Moreover, the switch and controller selection also pose a great challenge to switch migration. While some researchers suggested random selection of switch and controller selected from neighbours, other researchers suggested the selection of under-utilized controller, but no clear selection scheme for multiple controllers. Hence, migration efficiency during the reassignments of controllers and switches should be considered. Similarly, Das *et al.* [45] stated that though mapping multiple controllers per switch is beneficial in terms of security and resiliency, it also increases the network complexity in terms of control traffic. Consequently, security and resiliency trade-off should be investigated by exploring the impact of control latency and load balancing on multiple controllers.

VI. CONCLUSION

CPP constitutes a critical challenge faced in the realm of the SDN when multiple controllers are deployed in the management of the networks and to achieve good QoS. Finding the optimal number of controllers and their location in the SDN networks requires good decision making and good planning. This paper reviewed, analyzed, and presented some of the important CPP approaches that have been proposed and developed over the years. The analyses were based on the solutions offered in each, the algorithms employed, and the challenges or limitations that exist. The objective is to bring together these developments and to provide future researches with directions. The findings of the analysis show that several CPP techniques exist such as cluster-based, LP/QP, bio-inspired, genetic algorithms, heuristic-based and greedy algorithm, simulated annealing, etc. while ML-based is developing. Each of these strategies has its objective, strengths, application, and weaknesses. Also, few of the CPP strategies incorporated a single objective while most employed multi-objective in their optimization model. Several performance metrics as shown in Fig. 2 were considered to realize the CPP goal and to deliver important QoS. Moreover, almost all the CPP strategies were designed for large-scale SDN and specific network types such as SDN-based WAN or CN, satellite, optical, WSN (SDWSN). Implementations were only possible through simulations and several topologies were used.

Several challenges were also identified which were provided as possible research directions such as the need for efficient CPP algorithm, cost-ware and energy-aware algorithms, and so on. Also, existing approaches are only based on SDWAN and less attention is paid on other paradigms such as SDWSN and so on. Several CPP schemes were found to be achieved in isolation without considering solutions to other problems such as security, and resource/request scheduling except in [13] where the attack-aware solution was considered. Consequently, this is why an effective and efficient CPP approach is yet to be realized, hence this calls for more research to be carried out in this regard. Also, more CPP scheme based on ML should be explored considering the effectiveness of the technique. Our future work is, therefore, centered on designing a dynamic, efficient, and energy-aware CPP algorithm in the SDWSN perspective.

REFERENCES

[1] M. Khorramizadeh and V. Ahmadi, "Capacity and load-aware software-defined network controller placement in heterogeneous environments," *Comput. Commun.*, vol. 129, pp. 226–247, Sep. 2018.

[2] K. S. Sahoo, D. Puthal, M. S. Obaidat, A. Sarkar, S. K. Mishra, and B. Sahoo, "On the placement of controllers in software-Defined-WAN using meta-heuristic approach," *J. Syst. Softw.*, vol. 145, pp. 180–194, Nov. 2018.

[3] A. A. Ateya, A. Muthanna, A. Vybornova, A. D. Algarni, A. Abuarqoub, Y. Koucheryavy, and A. Koucheryavy, "Chaotic salp swarm algorithm for SDN multi-controller networks," *Eng. Sci. Technol., Int. J.*, vol. 22, no. 4, pp. 1001–1012, Aug. 2019.

[4] T. Das and M. Gurusamy, "Resilient controller placement in hybrid SDN/legacy networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.

[5] S. Wu, X. Chen, L. Yang, C. Fan, and Y. Zhao, "Dynamic and static controller placement in software-defined satellite networking," *Acta Astronautica*, vol. 152, pp. 49–58, Nov. 2018.

[6] B. Zhang, X. Wang, and M. Huang, "Multi-objective optimization controller placement problem in Internet-oriented software defined network," *Comput. Commun.*, vol. 123, pp. 24–35, Jun. 2018.

[7] F. J. Ros and P. M. Ruiz, "On reliable controller placements in software-defined networks," *Comput. Commun.*, vol. 77, pp. 41–51, Mar. 2016.

[8] 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.

[9] A. Jalili, M. Keshtgari, R. Akbari, and R. Javidan, "Multi criteria analysis of controller placement problem in software defined networks," *Comput. Commun.*, vol. 133, pp. 115–128, Jan. 2019.

[10] B. P. R. Killi and S. V. Rao, "Towards improving resilience of controller placement with minimum backup capacity in software defined networks," *Comput. Netw.*, vol. 149, pp. 102–114, Feb. 2019.

[11] B. Isong, I. Mathebula, and N. Dladlu, "SDN-SDWSN controller fault tolerance framework for small to medium sized networks," in *Proc. 19th IEEE/ACIS Int. Conf. Softw. Eng., Artif. Intell., Netw. Parallel/Distrib. Comput. (SNPD)*, Busan, South Korea, Jun. 2018, pp. 43–51.

[12] 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.

[13] D. Santos, A. de Sousa, and C. M. Machuca, "Robust SDN controller placement to malicious node attacks," in *Proc. 21st Conf. Innov. Clouds, Internet Netw. Workshops (ICIN)*, Feb. 2018, pp. 1–8.

[14] T. Yuan, X. Huang, M. Ma, and J. Yuan, "Balance-based SDN controller placement and assignment with minimum weight matching," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2018, pp. 1–6.

[15] J. Lu, Z. Zhang, T. Hu, P. Yi, and J. Lan, "A survey of controller placement problem in software-defined networking," *IEEE Access*, vol. 7, pp. 24290–24307, 2019.

[16] M. He, P. Kalmbach, A. Blenk, W. Kellerer, and S. Schmid, "Algorithm-data driven optimization of adaptive communication networks," in *Proc. IEEE 25th Int. Conf. Netw. Protocols (ICNP)*, Oct. 2017, pp. 1–6.

[17] B. Heller, R. Sherwood, N. McKeown, "The controller placement problem," in *Proc. ACM SIGCOMM Workshop Hot Topics Softw. Defined Netw.*, HotSDN, Dec. 2012, pp. 7–12.

[18] M. Karakus and A. Durresi, "A survey: Control plane scalability issues and approaches in software-defined networking (SDN)," *Comput. Netw.*, vol. 112, pp. 279–293, Jan. 2017.

[19] 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.

[20] Y. Zhang, L. Cui, W. Wang, and Y. Zhang, "A survey on software defined networking with multiple controllers," *J. Netw. Comput. Appl.*, vol. 103, pp. 101–118, Feb. 2018.

[21] Y. E. Oktian, S. Lee, H. Lee, and J. Lam, "Distributed SDN controller system: A survey on design choice," *Comput. Netw.*, vol. 121, pp. 100–111, Jul. 2017.

[22] G. Wang, Y. Zhao, J. Huang, and W. Wang, "The controller placement problem in software defined networking: A survey," *IEEE Netw.*, vol. 31, no. 5, pp. 21–27, 2017.

[23] A.K. Singh, S. Srivastava, "A survey and classification of controller placement problem in SDN," *Int. J. Netw. Manage.*, vol. 28, no. 3, p. e2018, 2018.

[24] J. Hollinghurst, A. Ganesh, and T. Bauge, "Controller placement methods analysis," in *Proc. 6th Int. Conf. Inf. Commun. Manage. (ICICM)*, Oct. 2016, pp. 239–244.

[25] I. O. Adebayo, M. O. Adigun, and P. Mudali, "Feature selection strategies for the controller placement problem in SDNs: A review," in *Proc. Int. Conf. Adv. Big Data, Comput. Data Commun. Syst. (icABCD)*, Aug. 2018, pp. 1–5.

[26] K. S. K. Liyanage, M. Ma, and P. H. J. Chong, "Controller placement optimization in hierarchical distributed software defined vehicular networks," *Comput. Netw.*, vol. 135, pp. 226–239, Apr. 2018.

[27] Y. Zhao, C. Liu, H. Wang, X. Fu, Q. Shao, and J. Zhang, "Load balancing-based multi-controller coordinated deployment strategy in software defined optical networks," *Opt. Fiber Technol.*, vol. 46, pp. 198–204, Dec. 2018.

[28] B. P. R. Killi, E. A. Reddy, and S. V. Rao, "Cooperative game theory based network partitioning for controller placement in SDN," in *Proc. 10th Int. Conf. Commun. Syst. Netw. (COMSNETS)*, Jan. 2018, pp. 105–112.

[29] B. P. R. Killi and S. V. Rao, "Link failure aware capacitated controller placement in software defined networks," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Jan. 2018, pp. 292–297.

[30] 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.

[31] S. Lange, S. Gebert, T. Zinner, P. Tran-Gia, D. Hock, M. Jarschel, and M. Hoffmann, "Heuristic approaches to the controller placement problem in large scale SDN networks," *IEEE Trans. Netw. Service Manage.*, vol. 12, no. 1, pp. 4–17, Mar. 2015.

[32] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, "Reliability-aware controller placement for software-defined networks," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2013, pp. 672–675.

[33] Y. Hu, W. Wendong, X. Gong, X. Que, and C. Shiduan, "On reliability-optimized controller placement for software-defined networks," *China Commun.*, vol. 11, no. 2, pp. 38–54, Feb. 2014.

[34] L. Liao and V. C. Leung, "Genetic algorithms with particle swarm optimization based mutation for distributed controller placement in SDNs," in *Proc. IEEE Conf. Netw. Function Virtualization Softw. Defined Netw. (NFV-SDN)*, Nov. 2017, pp. 1–6.

[35] S. Liu, R. Steinert, and D. Kostic, "Flexible distributed control plane deployment," in *Proc. NOMS-IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2018, pp. 1–7.

[36] L. F. Müller, R. R. Oliveira, M. C. Luizelli, L. P. Gaspary, and M. P. Barcellos, "Survivor: An enhanced controller placement strategy for improving SDN survivability," in *Proc. IEEE Global Commun. Conf.*, Dec. 2014, 1909-1915.

[37] P. Vizarreta, C. M. Machuca, and W. Kellerer, "Controller placement strategies for a resilient SDN control plane," in *Proc. 8th Int. Workshop Resilient Netw. Design Model. (RNDM)*, Sep. 2016, pp. 253–259.

[38] J. Liu, J. Liu, and R. Xie, "Reliability-based controller placement algorithm in software defined networking," *Comput. Sci. Inf. Syst.*, vol. 13, no. 2, pp. 547–560, 2016.

[39] M. Tanha, D. Sajjadi, R. Ruby, and J. Pan, "Capacity-aware and delay-guaranteed resilient controller placement for software-defined WANs," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 3, pp. 991–1005, Sep. 2018.

[40] A. Sallahi and M. St-Hilaire, "Optimal model for the controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 19, no. 1, pp. 30–33, Jan. 2015.

[41] 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.

[42] Y. Hu, T. Luo, N. C. Beaulieu, and C. Deng, "The energy-aware controller placement problem in software defined networks," *IEEE Commun. Lett.*, vol. 21, no. 4, pp. 741–744, Apr. 2017.

[43] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "Efficient controller placement and reelection mechanism in distributed control system for software defined wireless sensor networks," *Trans. Emerg. Telecommun. Technol.*, vol. 30, no. 6, p. e3588, Jun. 2019.

[44] T. Zhang, P. Giaccone, A. Bianco, and S. De Domenico, "The role of the inter-controller consensus in the placement of distributed SDN controllers," *Comput. Commun.*, vol. 113, pp. 1–13, Nov. 2017.

[45] T. Das, V. Sridharan, and M. Gurusamy, "A survey on controller placement in SDN," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 472–503, 1st Quart., 2020.

[46] A. Kumari and A. Singh Sairam, "A survey of controller placement problem in software defined networks," 2019, *arXiv:1905.04649*. [Online]. Available: http://arxiv.org/abs/1905.04649

[47] G. Wang, Y. Zhao, J. Huang, and Y. Wu, "An effective approach to controller placement in software defined wide area networks," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 1, pp. 344–355, Mar. 2018.
[48] T. Wang, F. Liu, and H. Xu, "An efficient online algorithm for dynamic SDN controller assignment in data center networks," *IEEE/ACM Trans. Netw.*, vol. 25, no. 5, pp. 2788–2801, Oct. 2017.
[49] H. K. Rath, V. Revoori, S. M. Nadaf, and A. Simha, "Optimal controller placement in software defined networks (SDN) using a non-zero-sum game," in *Proc. IEEE Int. Symp. World Wireless, Mobile Multimedia Netw.*, Jun. 2014, pp. 1–6.
[50] A. Ksentini, M. Bagaa, T. Taleb, and I. Balasingham, "On using bargaining game for optimal placement of SDN controllers," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.
[51] V. Huang, G. Chen, P. Zhang, H. Li, C. Hu, T. Pan, and Q. Fu, "A scalable approach to SDN control plane management: High utilization comes with low latency," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 2, pp. 682–695, Jun. 2020.
[52] H. I. Kobo, A. M. Abu-Mahfouz, and G. P. Hancke, "A survey on software-defined wireless sensor networks: Challenges and design requirements," *IEEE Access*, vol. 5, pp. 1872–1899, 2017.
[53] H. Mostafaei, M. Menth, and M. S. Obaidat, "A learning automaton-based controller placement algorithm for software-defined networks," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2018, pp. 1–6.
[54] Y. Jimenez, C. Cervello-Pastor, and A. J. Garcia, "On the controller placement for designing a distributed SDN control layer," in *Proc. IFIP Netw. Conf.*, Jun. 2014, pp. 1–9.
[55] P. Xiao, W. Qu, H. Qi, Z. Li, and Y. Xu, "The SDN controller placement problem for WAN," in *Proc. IEEE/CIC Int. Conf. Commun. China (ICCC)*, Oct. 2014, pp. 220–224.
[56] P. Xiao, Z.-Y. Li, S. Guo, H. Qi, W.-Y. Qu, and H.-S. Yu, "A k self-adaptive SDN controller placement for wide area networks," *Frontiers Inf. Technol. Electron. Eng.*, vol. 17, no. 7, pp. 620–633, Jul. 2016.
[57] Q. Qin, K. Poularakis, G. Iosifidis, and L. Tassiulas, "SDN controller placement at the edge: Optimizing delay and overheads," in *Proc. IEEE INFOCOM-IEEE Conf. Comput. Commun.*, Apr. 2018, pp. 684–692.
[58] H. Kuang, Y. Qiu, R. Li, and X. Liu, "A hierarchical K-Means algorithm for controller placement in SDN-based WAN architecture," in *Proc. 10th Int. Conf. Measuring Technol. Mechatronics Autom. (ICMTMA)*, Feb. 2018, pp. 263–267.
[59] N. Cai, Y. Han, Y. Ben, W. An, and Z. Xu, "An effective load balanced controller placement approach in software-defined WANs," in *Proc. MILCOM - IEEE Mil. Commun. Conf. (MILCOM)*, Norfolk, VA, USA, Nov. 2019, pp. 361–366.
[60] A. K. Singh, S. Maurya, and S. Srivastava, "Varna-based optimization: A novel method for capacitated controller placement problem in SDN," *Frontiers Comput. Sci.*, vol. 14, no. 3, Jun. 2020, 143402.

BASSEY ISONG (Member, IEEE) received the B.Sc. degree in computer science from the University of Calabar, Nigeria, in 2004, the M.Sc. degrees in computer science and software engineering from the Blekinge Institute of Technology, Sweden, in 2008 and 2010, respectively, and the Ph.D. degree in computer science from the North-West University, in 2014. His research interests include and are not limited to software engineering, security, software-defined wireless sensor network, low power wide area networks, cloud computing, the Internet of Things, machine learning, and big data. He is a Senior Lecturer with the Department of Computer Science and a faculty member of the North-West University, Mafikeng Campus, South Africa. He is also a member of the IEEE Computer, Communication, and Education Societies as well as the ACM.

REORAPETSE RAMOLITI SAMUEL MOLOSE was born in Mafikeng, South Africa, in March 1995. He received the B.Sc. and B.Sc. (Hons.) degrees in computer science from North-West University, Mafikeng Campus, South Africa, in 2017 and 2018, respectively, where he is currently pursuing the M.Sc. degree with the Computer Science Department. His research interests include SDN, the Internet of Things, cloud computing, and machine learning.

ADNAN M. ABU-MAHFOUZ (Senior Member, IEEE) received the M.Eng. and Ph.D. degrees in computer engineering from the University of Pretoria. He is currently the Centre Manager of the Emerging Digital Technologies for 4IR (EDT4IR) Research Center, Council for Scientific and Industrial Research (CSIR); a Professor Extraordinaire at the Tshwane University of Technology; a Visiting Professor with the University of Johannesburg; and an Extraordinary faculty member at the University of Pretoria. His research interests are wireless sensor and actuator network, low power wide area networks, software-defined wireless sensor network, cognitive radio, network security, network management, and sensor/actuator node development. He has participated in the formulation of many large and multidisciplinary R&D successful proposals (as Principal Investigator or main author/contributor). He is the Founder of the Smart Networks collaboration initiative that aims to develop efficient and secure networks for future smart systems, such as smart cities, smart grid, and smart water grid. He is a member of many IEEE Technical Communities. He is an Associate Editor at IEEE Access, the IEEE Internet of Things, and the IEEE Transaction on Industrial Informatics.

NOSIPHO DLADLU received the B.Sc. (Hons.) and M.Sc. degrees in computer science from North-West University, Mafikeng Campus, South Africa, in 2011 and 2014, respectively, where she is currently pursuing the Ph.D. degree. She is also a Lecturer with the Department of Computer Sciences, North-West University, Mafikeng Campus. Her research interests include cloud computing, software engineering, the Internet of Things, and software-defined networks.