Received October 21, 2017, accepted January 26, 2018, date of publication March 5, 2018, date of current version March 28, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2805842

Load Balancing Mechanisms in the Software Defined Networks: A Systematic and Comprehensive Review of the Literature

ALI AKBAR NEGHABI[1], NIMA JAFARI NAVIMIPOUR[2], MEHDI HOSSEINZADEH[3,4], AND ALI REZAEE[1]

[1]Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran 1477893855, Iran
[2]Department of Computer Engineering, Tabriz Branch, Islamic Azad University, Tabriz 1477893855, Iran
[3]International Campus, Iran University of Medical Sciences, Tehran 1477893855, Iran
[4]Department of Computer Science, University of Human Development, Sulaimaniyah 0778-6, Iraq

Corresponding author: Mehdi Hosseinzadeh (hosseinzadeh.m@lums.ac.ir)

ABSTRACT With the expansion of the network and increasing their users, as well as emerging new technologies, such as cloud computing and big data, managing traditional networks is difficult. Therefore, it is necessary to change the traditional network architecture. Lately, to address this issue, a notion named software-defined network (SDN) has been proposed, which makes network management more conformable. Due to limited network resources and to meet the requirements of quality of service, one of the points that must be considered is load balancing issue that serves to distribute data traffic among multiple resources in order to maximize the efficiency and reliability of network resources. Load balancing is established based on the local information of the network in the conventional network. Hence, it is not very precise. However, SDN controllers have a global view of the network and can produce more optimized load balances. Although load balancing mechanisms are important in the SDN, to the best of our knowledge, there exists no precise and systematic review or survey on investigating these issues. Hence, this paper reviews the load balancing mechanisms which have been used in the SDN systematically based on two categories, deterministic and non-deterministic. Also, this paper represents benefits and some weakness regarded of the selected load balancing algorithms and investigates the metrics of their algorithms. In addition, the important challenges of these algorithms have been reviewed, so better load balancing techniques can be applied by the researchers in the future.

INDEX TERMS Load balancing, review, SDN, software defined networks, systematic.

I. INTRODUCTION

At first, the concept of Software Defined Network (SDN) is proposed by Stanford University [1]. SDN is a dynamic, cost-effective, manageable, and adaptable network architecture [2], [3]. It decouples the control and the data plane [4], [5]. The control layer including a centralized SDN controller that routing of packets is one of its responsibilities [6]. The data plane presents infrastructure layer that consists of a set of connected forwarding elements, such as SDN switches. It is responsible to make the effective routing orders and to transmit the information [7]. Some important components in the SDN are the controllers and OpenFlow protocol [8]. A controller is an application that becomes a strategic point in SDN. The controller manages control flows on switches or routers through the southbound Application Programming Interfaces (APIs) such as OpenFlow protocol and manages applications and business processes through the northbound APIs such as Representational State Transfer (REST) to implement smart networks [9]. OpenDaylight controller is one of the controllers that can run on all operating systems and hardware as well as it supports Java [10], [11]. OpenFlow is a standard communication protocol on a network that communicates the control layer and the forwarding layer of an SDN architecture. OpenFlow allows direct access and manipulation of forwarding plane on a network device such as virtual or physical switches and routers [12].

Load balancing is a technique to divide the workload onto multiple resources in order to avoid overload on any of the resources [13]. Maximizing throughput, minimizing response time and optimizing traffic are some of the load balancing

goals [14]. Conventional networks have not a global view of the network. Hence, load balancing methods in traditional networks have not precise [15]. But, SDN load balancing methods are more accurate and have higher performance. In SDN associated research, load balancing issue is one of the most important issues because of industry concerns [16].

Although, to the best of authors' knowledge, in spite of the fact that the load balancing mechanisms are too important in the SDN, there is not any wide-ranging and complete systematic review in this field. Hence, in this article, we investigate the SDN architecture and the OpenFlow protocol and then discuss the load balancing problem in the SDN network. Furthermore, the most significant qualitative parameters for load balancing in the SDN will be described. The main goal of this paper is to survey the current mechanisms, then compares the features of the selected mechanisms, and finally, defines specified common load balancing mechanisms in the SDN and outlines the categories of issues that would be considered. In a nutshell, the main contributions of this paper are as follows:

- Presenting a review of the challenges related to SDN that are discussed the use of load balancing;
- Providing a comprehensive systematic review of the current mechanisms for load balancing and the approach in which these have been applied to SDN;
- Exploring the future research directions and the role that load balancing can play in the SDN;
- Outlining the key research directions where future works can optimize the effectiveness of load balancing methods in the SDN.

The remainder of this paper is organized as follows. After the introduction, backgrounds of SDN and load balancing are provided in Section II. Section III discusses some important related work. The research methodology and papers selection mechanisms are provided in Section IV. Section V discusses load balancing mechanisms in the SDN and classifies them, also provides the taxonomy and comparison of the discussed mechanisms. Results and comparison are discussed in Section VI. Section VII discusses many open issues. Finally, Section VIII discusses conclusion and limitation of this paper.

II. BACKGROUND

In this section, the structure of SDN architecture has been studied and additionally, the principal benefits of using SDN have been explained. The concept and structure of load balancing have been divided into two major categories which include: load balancing in the IP network and load balancing in the SDN network [17]. Lastly, the parameters that affect the efficiency of load balancing have been described.

A. SDN ARCHITECTURE

One of the novel network architecture is the SDN architecture. It provides a network environment that an SDN controller centrally manages the network. The most appropriate standard to implement SDN architecture is OpenFlow protocol. Through the use of controllers, SDN architecture with the OpenFlow protocol offers better strategy in comparison to conventional networks for flows processing by network operators.

SDN represents a real revolution in the network's world. Devices of traditional network contain both the control and the data plane. But, SDN refers to a network architecture that introduces a clear separation between the data plane (forwarding plane) and the control plane. Control plane is the brain of the network that responsible for managing the network centrally, and the data plane is underlying infrastructure such as switches and routers [18], [19]. As shown in Fig. 1, SDN architecture is divided into three major planes by Open Networking Foundation (ONF):

- Application Layer: An SDN application layer has one or more end-user applications which capture an abstract view of the network so that they can demonstrate their internal decision-making process. The API is used by the programmer to implement their applications and communication with the controller is called the northbound API [18], [20].
- Control Layer: The intermediate level constitutes the control plane. It is a Network Operating System (NOS) that controls network platform [21]. The task of the control plane is twofold: on the one hand, it is responsible for the administration of the switches, which will be instructed on the routing methods of the packets in transmission, on the other hand, it must serve to the higher level via creating an abstract and centralized vision of the underlying infrastructure [22]. The controller uses the southbound API that to communicate with network devices. The OpenFlow protocol is the most important of southbound API [18], [19], [21]. The load balancer is a component of the SDN controller which is located in a logical central point of decision and load balancing algorithms can be installed on it.
- Data Layer: The lowest layer of architecture can be called the Infrastructure Layer. This level represents the data plane of the network and consists of physical and/or virtual devices such as switches. The most important function that a switch must perform is that to forwarding packets according to a certain set of rules that are specified by the SDN controller. The SDN controller is responsible for to definition and installation of these rules on the flow table of switches [23].

B. LOAD BALANCING IN SDN

Mainly, in distributed systems to improve overall cluster performance, load balancing technology is utilized [24]–[26]. Load balancing can be implemented in software or in a physical equipment. It is responsible for distributing the load between several resources of the same type [27]. Load balancing has different methods, these methods can be static, dynamic or a combination of both. Having prior information of the system is an essential feature of these static methods. In these methods, the rule is directly programmed in the load

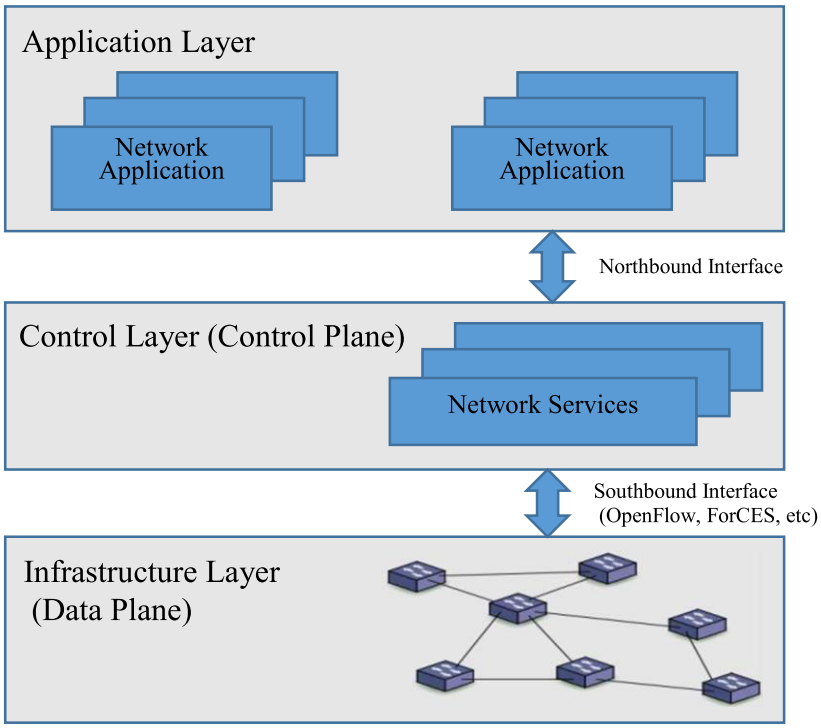

FIGURE 1. An Overview of SDN architecture with its main planes: data, control and application plane [21].

balancer, since the behavior of the user cannot be predicted, static load balancing methods can be inefficient in a network. The dynamic methods are more efficient than the static methods because the load is distributed dynamically according to some pattern programmed in the load balancer [27], [28]. A suitable load balancing helps in maximizing scalability, minimum response time, maximize throughput, minimizing resource consumption, avoiding overload of any single resource and so on. There are two main types of load balancing: load balancing in the IP network and in the SDN network [17], [29]. The Load Balancing Router (LBR) in the traditional IP-based network balances the load and if a new flow enters the network, it will initially go through LBR [17]. Subsequently, the LBR chooses a server according to current network status as target server of the new flow. Then, the IP address of the target server is inserted into the packet of the new flow. Later, using routing protocols, the packet will be transmitted to its target server via the calculated path. Alternatively, when selecting the destination server in the IP-based network, the routing decision is only made based on the network state at the time of the selection of the target server [17]. In the SDN network, to fulfill the required load balancing, the controller has to complete a series of Real-time Least loaded Server selections (RLSs). To specify the target server of a new flow, the RLS is applied and it is also used to compute a path leading to the target server while the new flow enters into a domain for the first time. Based on a real-time network situation, the RLS makes the forwarding decision for every new flow [17]. However, using RLS as a centralized controller in the SDN poses some problems such as bottlenecks of a single controller, responsiveness, reliability and poor scalability [30]–[33]. To overcome the mentioned problems, using multiple distributed controllers working together is a simple solution to fulfill the function of the logically centralized controller [17], [34].

C. LOAD BALANCING PARAMETERS

Some parameters are required to evaluate a load balancing algorithm and compare it with previous methods in order to specify the better load balancing algorithm and to recognize the advantages and disadvantages of it. These parameters call qualitative parameters. Articles use different qualitative parameters such as throughput, utilization, and latency. The most significant qualitative parameters for load balancing in the SDN are described as follows:

An average number of synchronizations per minute: It is the average number of controller state synchronization per minutes in SDN that uses distributed controllers [17].

Cumulative frequency: A performance index that supplies a measure of the accuracy of an algorithm has been introduced by Boero *et al.* [35] to ideally provide the exact amount of traffic in each queue to get load balancing. This parameter is computed at each time instant t as:

$$index^t = \frac{\sum_i (r_{qi}^t - r^{-t})^2}{4}, \quad t = 0, 1, \ldots. \tag{1}$$

Where r_{qi}^t is the measured output rate of queue qi and r^{-t} is the optimal queue rate at time instant t. index^t is

a distance between the current solution and the ideal one [35].

Degree of Load Balancing: It is a metric of uniformity of the load distribution among entities. There are multiple indexes that measure this metric such as Jain's fairness index [36], [37] and the arithmetic average for the coefficient of variation [19].

Energy Consumption: It is the amount of consumed energy in the network. Effective load balancing mechanism can reduce the energy consumption [38], [39].

Execution Time: It is the length of time that a program is running. Execution time can include migration time [40], routing time [41] and re-association time [37].

Forwarding Entries: Routers use forwarding table to decide to send the packet out. Decreasing the number of forwarding entries can be effective in saving memory resources [42].

Guaranteed Bit Rate (GBR): It is one of the Quality of Service (QoS) parameters in Long Term Evolution (LTE) networks for guaranteeing the bandwidth of the bearer [36].

Latency: It is the time required to forward a packet across a network. There are many kinds of latency such as traffic delivery latency [43] and communication latency [44].

Migration Cost: It consists two main costs: the message exchanging cost and load cost. Some messages must be transmitted between the controllers, for doing switch migration such as migration request, role request, and asynchronous messages. Message exchanging cost is the cost of exchanging these messages between controllers [40], [45]. When a switch sk is migrated from controller c_i to controller c_j, load cost is described by the following equation [40]:

$$r_{LC} = \begin{Bmatrix} f_{s_k} d_{s_k c_j} - f_{s_k} d_{s_k c_i}, & f_{s_k} d_{s_k c_j} > f_{s_k} d_{s_k c_i} \\ 0, & f_{s_k} d_{s_k c_j} \leq f_{s_k} d_{s_k c_i} \end{Bmatrix} \quad (2)$$

Where f_{s_k} is the number of packet-in messages sent from switch s_k to controller c_i and $d_{s_k c_i}$ is the minimal path cost from switch s_k to controller c_i [40].

Overhead: Any composition of excessive or indirect computation time, memory, bandwidth, or other resources that are needed to carry out a particular task is overhead. There are different types of overhead such as communication overhead [46], the overhead of flow stealing [47], synchronization overhead [17] and flow statistics collection overhead [48].

Overload Ratio: Rangisetti and Tamma [36] have introduced a new cell overload definition named OverLoad Ratio (OLR), in order to include accurate load definition networks. In the LTE networks, OLR of a cell is defined in a specific period by employing its resource block utilization and QoS satisfaction of GBR User Equipments (UEs). For an instant, a specific cell with OLR $= 0$ indicates that in that specific cell all GBR UEs can get 100% of their organized GBR. Further, OLR $= 0.7$ indicates that GBR UEs can get only 30% of their configured GBR [36].

Packet Loss Rate: Packet loss happens when one or more packets of data do not reach their target. It usually results from network congestion. It is the percentage of packets lost regarding packets sent. Also, packet loss rate is the rate of packets loss [14], [49].

Peak Load Ratio: For route performance measurement, Xu *et al.* [48] have introduced the metric of Peak Load Ratio (PLR). First, the traffic load $f(e)$ of each link $e \in E$ is measured and then PLR is defined as:

$$\text{PLR} = max \{f(e)/c(e), e \in E\} \quad (3)$$

Percentage of matched deadline flows: This parameter represents the percentage of flows satisfying the deadline. Some flows may have deadline constraints, for example, the flow is useful if, and only if, it totally arrived at the target within the deadline [35].

Resource Utilization: It is a degree to which the resources of the network are utilized such as link, bandwidth, processor and memory utilization. Maximum resource utilization is provided by an acceptable load balancing algorithm [38], [50]–[52].

Response Time: It is defined by the interval that starts from accepting a request or job to responding to a request or task for server [52], [53].

Root Mean Squared Error (RMSE): It is a metric for assessing load balancing performance [54]. A better performance has a smaller RMSE. RMSE will be 0 if a load of all servers is the same [17].

Throughput: It is the quantity of data that has been correctly moved from one place to another during a certain period of time [55]–[57].

Workload: It is the amount of work to be done by the controller. In order to balance the workloads among controllers, load balancing approaches have been introduced [47].

III. RELATED WORK

A review of network update mechanisms in the SDN has been presented by Dan *et al.* [58]. This survey paper describes network update problem and summarily depicts the problems resulted from network update, in addition, expresses the solutions in the SDN paradigm. During the network updating process, four basic confusions such as forwarding black hole, forwarding loop, link congestion and network policy violation have particularly been examined. They also have studied the solutions of the aforementioned problems and have discussions of the limitations for network updating schedule. Finally, the discussion about the difficulties for solving various problems has accomplished. However, there is a gap for papers selection mechanism. Additionally, future works have not been properly described. Moreover, load balancing has not been investigated and most of the published articles in 2016 have not been mentioned.

Trois *et al.* [59] have collected all the important functions of SDN programming language as well as mapping and grouping the similar ones. A taxonomy has been provided enabling us to identify the functions defined by the SDN languages. The Feature-Oriented Domain Analysis (FODA) method has been used to create this taxonomy. The languages have been classified based on their programming paradigm,

the way their policies are defined, the way through which the flows are installed on switches, as well as the provided abstractions. In order to focus on their primary contributions, an evolutionary assessment of the prominent languages has been offered. A genealogy has also been depicted to demonstrate the relationships between these languages. However, there is a gap for discussion in papers selection mechanism and recently published papers such as [60]. Also, the problem of load balancing in the SDN has not been discussed.

Also, Rowshanrad *et al.* [61] have reviewed wired and wireless SDN architectures and different protocols and approaches utilized in these architectures such as OpenFlow, XMPP, OnePK, Openroad, SoftRAN, and SoftCell. A number of currently SDN available switches have been listed. Besides, SDN emulators and simulators such as EstiNet, Mininet, and NS-3 have compared. However, this paper does not contain the published papers in Border Gateway Protocol (BGP) and Open vSwitch Database Management Protocol (OVSDB). Also, this survey has written in a non-systematically manner and load balancing has not been investigated. Furthermore, open issues have not been well described and recently published papers have not been investigated.

Furthermore, Jarraya *et al.* [62] have surveyed the literature on SDN over the 2008-2013 period. In order to supply a deep and complete understanding of this paradigm, its associated technologies, its domains of application, also the main problems that should be addressed towards maintaining its achievements are investigated. They have provided a more comprehensive and up-to-date overview of SDN with focusing on more than one aspect while investigating most related research and recognizing predictable future research lines. They have defined SDN classifications and have defined a taxonomy of SDN. The suggested taxonomy provides a hierarchical view and categorizes the recognized problems and solutions in each layer: application, control, and infrastructure. However, this paper does not contain published papers in the 2014-2017 period and does not a systematical study. Also, load balancing issues in the SDN have not been discussed.

Moreover, Huang *et al.* [10] have presented a survey and research challenges for large-scale SDN testbeds. They have described an overview of SDN testbeds and five common implementations of large-scale SDN testbeds all over the world, consisting of Global Environment for Network Innovation (GENI) OpenFlow, OpenFlow in Europe Linking Infrastructure and Applications (OFELIA), Research Infrastructure for large-Scale network Experiments (RISE), OpenFlow @ Trans-Eurasian Information Network (OF@TEIN) and OpenLab which includes design objectives, deployment, key technologies, and experiments. They have compared the SDN testbeds based on objective and development, management and networking, slicing, network deployment parameters. However, this paper does not contain the published papers in Joint Open Lab SDN Network (JOLNET) and Community Connection (CoCo) testbeds. Also, it does not a systematical study as well as it has not papers selection mechanism and the problem of load balancing in the SDN have not been discussed.

Finally, reviewing the QoS parameters in SDN has been accomplished by Karakus and Durresi [63]. They have prepared the relevant research works depending on the types that are the most prominent methods in which QoS can benefit from the idea of SDN: resource reservation mechanisms, scheduling mechanisms, inter-domain routing approaches, queue management, network monitoring mechanisms, Quality of Experience (QoE)-aware mechanisms and other QoS-centric parameters such as QoS policy management and virtualization-based QoS provisioning. They have also discussed the QoS capabilities of OpenFlow protocol through reviewing its kinds along with some popular, open-source, and community-driven controller projects. In this study, the papers selection mechanism is not clear and future works have not been properly described. Moreover, load balancing problem and recently published papers have not been investigated.

It is important to consider that these surveys were not prepared a pure systematic literature-based review of the current load balancing techniques, future challenges, their classification, and the key role that load balancing could have in the SDN. By responding to each of these questions, this paper formalizes three questions in the next section to choose remarkable studies for evaluation and then accents the significance of load balancing mechanisms, present challenges, and future directions in SDN.

IV. ARTICLES SELECTION METHOD

This section provides a Systematic Literature Review (SLR) methodology which is proposed by [64] with a particular attention on studies relevant to load balancing mechanisms in the SDN to increase apprehending of them. An SLR first used in medicine fields [64] which offers a repeatable research method and needs to provide adequate details to carry out by other researchers [65]–[67]. In this section, the SLR is employed to carry out a wide-ranging and systematic study of the load balancing algorithms in the SDN. Researchers have proposed three research questions to deal with the key issues of load balancing in the SDN to emboss the incumbency of load balancing in the SDN. In the next section, these questions are formalized.

A. QUESTION FORMALIZATION

In this section, the most related issues and challenges in the field of load balancing in the SDN are identified, such as overloading, underloading, response time, costs, throughput, and possible balancing solutions. This study tries to answer the following research questions:

RQ1: What is the emphasis of load balancing in the SDN?

This question determines the number of SDN load balancing studies which have been published all the time to emphasize the significance of it in SDN.

RQ2: How much do the current methods meet the main metrics of load balancing?

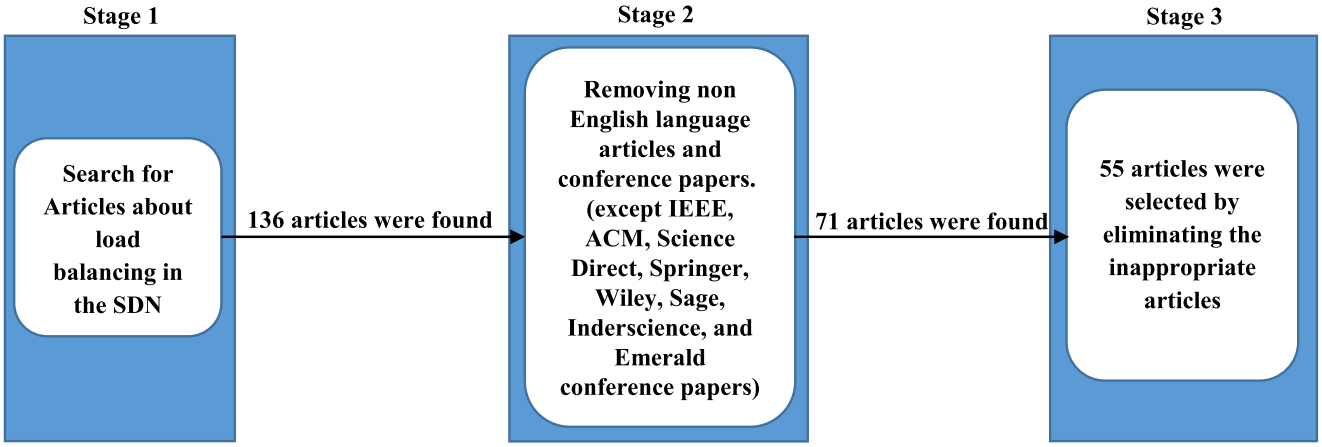

FIGURE 2. Filtering method for found articles.

This question targets at evaluating the existing load balancing approaches derived from the primary metrics in SDN.

RQ3: What problems and solutions can be specified regarding the load balancing in the coming years?

This question seeks to clarify the role of load balancing in the SDN, and identify the challenges and the techniques applied to guarantee the QoS.

B. ARTICLE SELECTION PROCESS

The article selection process is performed on three stages, including [66]:

- Automated keyword-based search.
- Selection of the article based on the title, abstract, and quality of the publication.
- Eliminated the inappropriate articles.

In the automated search based on keywords stage, the search process is performed using electronic searching on some popular academic databases such as IEEE explorer,[1] Sage,[2] Google Scholar,[3] ACM,[4] Wiley,[5] Inderscience,[6] Emerald,[7] Springer,[8] and Science Direct.[9] The following search string was defined by adding other spellings of the main elements to find relevant articles.

✓ (“Software Defined Network” OR “SDN”) AND (“Load” OR “Balancing”)

We found 136 articles from the journals, conference proceedings, patent, books, and thesis. These articles were published between 2013 up to 2017.

In the article selection based on the quality of the publisher stage, the search string is constrained by searching for conference papers and journal articles of IEEE, Sage, ACM, Wiley, Science Direct, Emerald, Springer, and Inderscience to guarantee that only high-quality publications and articles are selected for the review [68]. For possibility reasons, the papers which are not written in English are removed. Consequently, 71 articles are selected. Fig. 2 indicates an overview of the implemented process for identifying the articles in this study.

In the eliminated the inappropriate articles stage, a Quality Assessment Checklist (QAC) based on Kitchenham *et al.* [69] is developed where those articles emerged from the initial search are refined. After reading abstracts and searching keywords, we eliminated the inappropriate articles. Then, the entire body of the remaining papers was checked and those which were not related to our concerned area were also crossed out. After eliminated inappropriate articles, 55 studies were identified which are shown in Fig. 3 where 9% are related to ACM, 16% of the articles are related to Springer, 64% are related to IEEE, 2% are related to Sage, 9% are related to Science Direct and 0% are related to Inderscience, Wiley, and Emerald.

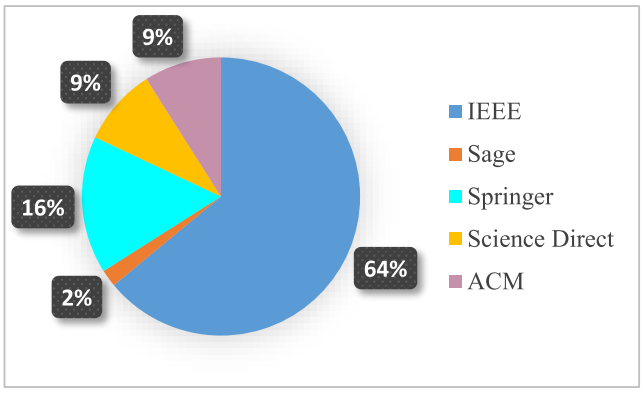

FIGURE 3. Percentage of published articles in any publication.

Selected papers are published between 2013 up to 2017. Fig. 4 shows the distribution of the articles by the year of

[1] http://ieeexplore.ieee.org
[2] http://journals.sagepub.com
[3] http://Scholar.google.com
[4] http://www.acm.org
[5] http://onlinelibrary.wiley.com
[6] http://www.inderscience.com
[7] http://www.emeraldinsight.com
[8] http://link.springer.com
[9] http://www.sciencedirect.com

TABLE 1. Details of the selected articles that used deterministic approaches.

Publisher	Year	Article	Author	Journal / Conference
IEEE	2015	A Traffic Load Balancing Framework for Software-Defined Radio Access Networks Powered by Hybrid Energy Sources	Han and Ansari [43]	IEEE/ACM Transactions On Networking
	2016	A Dynamic Load Balancing Method of Cloud-Center Based on SDN	Yong, et al. [72]	Network Technology And Application
Springer	2015	Software-Defined Networking-Based Resource Management: Data Offloading with Load Balancing in 5G HetNet	Duan, et al. [71]	EURASIP Journal on Wireless Communications and Networking
	2016	An SDN-Enhanced Load-Balancing Technique in the Cloud System	Kang and Choo [53]	The Journal of Supercomputing
	2017	Flow Stealer: Lightweight Load Balancing by Stealing Flows in Distributed SDN Controllers	Song, et al. [47]	Science China Information Sciences
		Load-Balancing Multiple Controllers Mechanism for Software-Defined Networking	Ma, et al. [75]	Wireless Personal Communications
		Two-tier Dynamic Load Balancing in SDN-Enabled Wi-Fi Networks	Lin, et al. [37]	Wireless Networks
Elsevier	2014	Improving the Performance of Load Balancing in Software-Defined Networks through Load Variance-Based Synchronization	Guo, et al. [17]	Computer Networks
	2017	LBBSRT: An Efficient SDN Load Balancing Scheme Based on Server Response Time	Zhong, et al. [52]	Future Generation Computer Systems
		QoS Aware Load Balance in Software Defined LTE Networks	Rangisetti and Tamma [36]	Computer Communications
Sage	2015	A Multicontroller Load Balancing Approach in Software-Defined Wireless Networks	Yao, et al. [70]	International Journal of Distributed Sensor Networks
ACM	2016	Dynamic Load Balancing of Local Mobility Anchors in Software Defined Networking Based Proxy Mobile IPv6	Raza, et al. [74]	Proceedings of the 10th International Conference on Ubiquitous Information Management and Communication

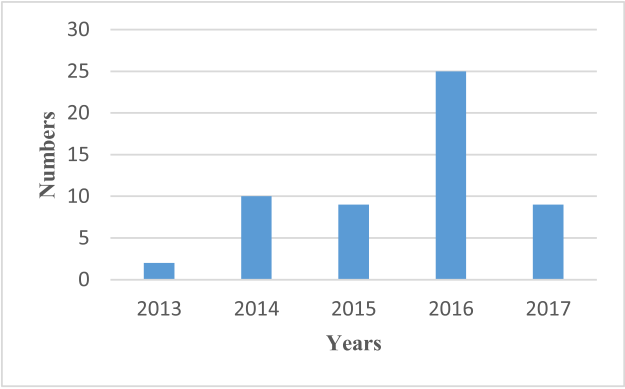

FIGURE 4. Distribution of the selected articles by year.

publication. For further detail analysis, we selected 19 articles based on full text and quality assessment to perform the more accurate analysis. 19 selected articles can be divided into two main classes including deterministic and non-deterministic approaches. Table 1 shows details of the selected articles that used deterministic approaches and Table 2 displays details of the selected articles that used nondeterministic approaches.

V. REVIEW OF THE SELECTED LOAD BALANCING MECHANISMS IN THE SDN

In this section, 19 selected articles based on the mentioned criteria will be reviewed. For this reason, techniques and basic properties of each paper and their differences, advantageous and disadvantageous will be described and explained. The surveyed papers use various methods such as switch migration, re-routing, approximation, greedy and so on. The best criterion that can be used for categorization is the deterministic and non-deterministic criteria that establish a logical relationship between these papers. Hence, the proposed methods in the literature have been classified into two major distinct groups including deterministic and non-deterministic approaches. The non-deterministic category includes greedy, approximate and heuristic methods so other methods are in the deterministic category. Classification of methods and their definitions are illustrated in Fig. 5. In Section A and B, these methods and their examples are discussed. Also, the used algorithm, advantages, and disadvantage in each category are reviewed.

A. DETERMINISTIC APPROACHES

A deterministic approach always produces the same output for a specific input. Its processes are often described by

TABLE 2. Details of the selected articles that used non-deterministic approaches.

Publisher	Year	Article	Author	Journal / Conference
IEEE	2014	Load Balancing for Multiple Traffic Matrices Using SDN Hybrid Routing	Zhang, et al. [42]	15th International Conference on High Performance Switching and Routing
	2015	Design of a Load-Balancing Middlebox Based on SDN for Data Centers	Tu, et al. [76]	The International Workshop of Software-Defined Data Communications and Storage
	2016	A Novel Load Balancing Strategy of Software-Defined Cloud/Fog Networking in the Internet of Vehicles	He, et al. [44]	China Communications
	2017	A Switch Migration-Based Decision-Making Scheme for Balancing Load in SDN	Wang, et al. [40]	IEEE Access
Springer	2014	A Genetic-Based Load Balancing Algorithm in OpenFlow Network	Chou, et al. [19]	Advanced Technologies, Embedded and Multimedia for Human-centric Computing
Elsevier	2016	BeaQoS: Load Balancing and Deadline Management of Queues in an OpenFlow SDN Switch	Boero, et al. [35]	Computer Networks
	2017	Partial Flow Statistics Collection for Load-Balanced Routing in Software Defined Networks	Xu, et al. [48]	Computer Networks

differential equations. Also, the output of the model is completely specified by the values of the parameters and the primary situations. In Section 1, the selected deterministic approaches are discussed and in Section 2, their summaries are presented. Most of the load balancing methods in the SDN have used deterministic approach.

1) OVERVIEW OF THE SELECTED DETERMINISTIC APPROACHES

The logically centralized controller in large-scale SDN networks typically includes multiple distributed controllers. Current multiple controllers synchronization methods, which can cause undesirable situations, are according to periodic synchronization. For example, frequent synchronizations can cause excessive controller synchronization overhead. State desynchronization among controllers during the interval between two consecutive synchronizations might cause to black holes and forwarding loops. Guo *et al.* [17] have proposed a controller state synchronization method named Load Variance-based Synchronization (LVS), in order to enhance the performance of load balancing in the multi-controller multi-domain SDN network. In comparison to periodic synchronization based methods, LVS-based methods performed actual state synchronizations among controllers while a load of a server exceeds a specified threshold, which considerably minimizes the synchronization overhead of controllers. The experimental results have proved that LVS obtains proper load balancing performance and loop-free forwarding with less synchronization overhead, in comparison to existing methods. However, two proposed LVS-based methods have not evaluated in a real testbed. Also, energy consumption and latency of the method have not been evaluated.

Furthermore, the increasing complexity of the wireless networks (i.e., 5G and wireless sensor networks) turns the network control and coordination into a dilemma. The future wireless networks require accurate separation of the control and data planes in addition to SDN method to handle the explosive rise in the traffic of the mobile data. Sticking to a single controller in future wireless networks causes a potential scalability problem. To manage the large wide-area wireless network, where the load balancing problem of the multi-controller needs to be resolved, the idea of using multiple controllers has been addressed. A multi-controller load balancing method in software-defined wireless networks called hybrid flow has been proposed by Yao *et al.* [70]. In this algorithm, the load balancing is carried out by distribution and centralization techniques. The network is divided into a number of clusters that consisting of several switches and cluster controllers, as well as a global controller. They have designed a double threshold approach to determine overload on cluster controllers and perform intra-cluster load balancing in clusters or global load balancing. Each cluster controller is responsible for deciding how to send packets and the global controller is responsible for load balancing among clusters. If the load of a cluster controller exceeds the threshold, load balancing is performed by transferring the load from an overload cluster controller to other controllers of the same cluster but if the load exceeds the cluster capacity, the overload cluster controller sends a request to the global controller to perform global load balancing so that the global controller distributing flows among clusters.

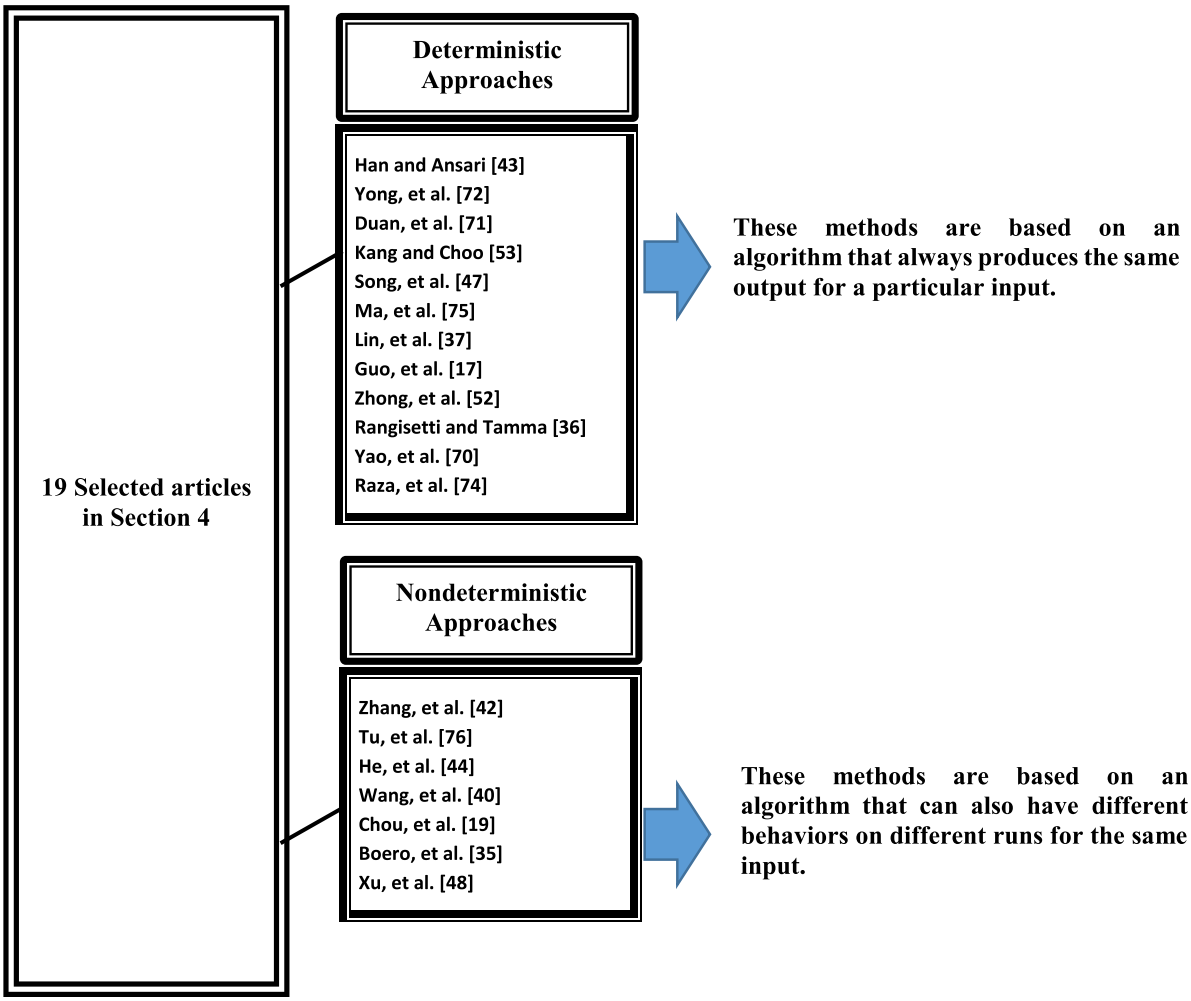

FIGURE 5. Classification of the load balancing methods in the SDN.

Simulation results have demonstrated that according to the proposed method compared with the balance flow method, the working load is relieved on the super controller and the load jitter of the multi-controller load is also reduced in a single cluster. However, it suffers from high complexity, due to using multi-controllers. Moreover, the overhead of the algorithm and its throughput have not been investigated.

On the other hand, the spectrum performance is improved by the small cells and the capacity of mobile networks could be expanded using this method. Thanks to energy harvesting technology, Base Stations (BSs) can be fueled by green energy so that they consume less on-grid power. To exploit the full capacity of Small Cell Base Stations (SCBSs), traffic load balancing seems essential for mobile networks with high BS density. Han and Ansari [43] have proposed a traffic load balancing model making an attempt to make a balance between network utilities, for example, the average traffic delivery latency, and the consumption of the green energy. The proposed model, as a virtually distributed algorithm, may be put into practice to reduce the communication overheads between users and BSs. According to the simulation findings, the proposed framework enables a trade-off which can be adjusted between the on-grid power consumption and the average traffic delivery latency. Furthermore, a large amount of on-grid power can be saved just at the cost of only a small increase in the average traffic delivery latency, but, since this method uses a single controller, it suffers from low scalability, availability and system bottleneck. It also increases a small amount of average traffic delivery latency. Furthermore, the throughput of the method has not been investigated.

Besides, efficient resource management is regarded as a serious challenge for upcoming 5G networks because the data traffic and the co-existence of various radio access technologies are increasingly prevailing. Duan *et al.* [71] have introduced SDN-based resource management algorithms for the upcoming cellular network. To do so, they follow three objectives: i) alleviate spectrum lack in relation to efficiently offloading traffic over the Wi-Fi network, ii) address the network congestion caused by balancing loads across multiple

cells and iii) attain the above mentioned objectives while considering the regular network situations as well as the end user QoS desires. To fulfill this objective, they have introduced an SDN-based partial load balancing and data offloading algorithms. Eventually, the performance of the proposed algorithm was analyzed through a real network model. Moreover, an analytical framework was also introduced to quantify the SDN-based data processing and forwarding delay. Experiments and system-level simulations have indicated that, in comparison with the baseline algorithms, the proposed algorithm improves the equilibrium extent, network stability, and throughput significantly. However, other performance parameters such as route delay and packet loss ratio have not been evaluated. Since it uses a single controller, it suffers from low scalability, low availability, and system bottleneck.

Also, Yong *et al.* [72] have proposed an SDN-based dynamic Load Balance solution (SDN-LB) with applying the SDN technology to the cloud data center and solve the load balance problem by employing SDN architecture. They have utilized the Plug-n-Server [73] as an SDN architecture. The Plug-n-Server includes three main parts: the underlying objective composite by servers and clients; OpenFlow switch network; SDN controller and decision platform. The controller contains four modules: traffic detection module, dynamic load scheduling module, load calculation module and flow management module. Traffic detection module is used for dynamic traffic monitoring and statistics; load calculation module aims to estimate the load distribution of the cloud environment. In dynamic load scheduling module, they have proposed a new hybrid load balancing algorithm to achieve high-performance load balancing for cloud center; flow management module responsibility is deployment load balance strategy based on hybrid load balance algorithm. The performance of SDN-LB has assessed in the comparison with other three versions of a state of the art hash-based solution via simulations. The simulation results have proven that SDN-LB achieves higher throughput than traditional methods. This method uses a single controller, for this reason, it has low scalability, low availability and system bottleneck. Furthermore, latency and utilization metrics have not been considered. Also, uniformity of the load distribution among servers have not been measured.

The majority of web services and sites are hosted by diverse types of cloud services, and such systems require efficient load-balancing policies to order some level of QoS which is possible through choosing multiple clouds. Kang and Choo [53] have introduced an SDN-enhanced InterCloud Manager (S-ICM) that assigns network flows in the cloud environment. The proposed approach contains two main parts, monitoring and decision making. For monitoring, S-ICM utilizes SDN control message that observes and collects data, and decision-making is made in accordance with the measured network delay of packets. Measurements are also employed to evaluate S-ICM and to compare it with a round robin tasks allocation where the workload is distributed via a Honeybee Foraging Algorithm (HFA). The evaluation results have shown that in term of avoiding the system saturation, S-ICM is better than HFA and round robin under the heavy load formula. Measurements are also employed to examine whether a simple queueing plan can be employed to predict the efficiency of the system for many clouds being operated under round robin scheduling strategy, and finally, they have proven the validity of the theoretical approximation. But, S-ICM generates additional control messages continuously into the whole network. In addition, uniformity of the load distribution among servers and throughput has not been measured.

Additionally, an SDN based solution has been proposed by Raza, et al. [74] to provide load balancing among mobility anchors based on the PMIPv6 domain. Proxy Mobile IPv6 (PMIPv6) is an IP mobility protocol where local mobility anchor is involved in control as well as data communication. The PMIPv6 standard enables the possibility of having multiple mobility anchors to reduce the load on a mobility anchor and avoid a single point of failure. In the proposed solution, a mobility controller acts as a central control entity and performs load monitoring. The controller moves the traffic from highly loaded mobility anchor to less loaded mobility anchor. Analytical modeling based on performance assessment results have shown that while the load balancing is being accomplished, the proposed scheme reduces the disruption duration of uplink and downlink traffic. However, this paper has not included the algorithms to perform load detection and mobile node selection. Moreover, this scheme suffers from low scalability, low availability and system bottleneck due to using a single controller. Also, throughput and degree of load balancing metrics have not been considered.

Switch migration commonly utilized by load balancing methods, which dynamically adjusts the mapping between controllers and switches based on controller workloads. Generally, switch migration-based methods face some challenges under the burst traffic due to their overhead and longer detection periods. Song *et al.* [47] have presented the flow stealer which is a lightweight load balancing technique for distributed SDN controllers. Flow stealer utilizes a low-cost flow-stealing approach, in which idle controllers share workloads temporarily with overloaded controllers. The flow-stealing technique can respond to variations of network traffic more rapidly, and the frequency of switch migration is eventually reduced. Furthermore, flow stealer joins both flows stealing and switch migration to adapt and burst the traffic and long-term traffic variations. Simulation results have shown that flow stealer efficiently balances the workloads between controllers, especially under burst traffic, whereas energy consumption and latency of the method have not been evaluated. Also, the complexity of the proposed method is high due to the using multiple controllers.

Furthermore, in order to use the advantage of SDN flexibility, Zhong *et al.* [52] have proposed a Load Balancing scheme Based on Server Response Times, named LBBSRT. LBBSRT is under the SDN architecture employing the controller to

attain ultimately the real response time of each server so that it can choose a server with more stable response time to solve some load balancing issues in the traditional networks such as upper deployment costs and lower efficiency. Simulation experiments have shown that the proposed scheme process demands with a low average server response time and achieves a good load balancing. Also, the implementation of the proposed method is simple and efficiently solves the load balancing issue with low cost and proper scalability, but, the authors have not taken into consideration the energy saving issue. Also, this method uses one controller, therefore, it suffers from low availability, low scalability, and system bottleneck.

Also, Ma *et al.* [75] have proposed a load-balancing mechanism for using in a multiple-controller SDN that implements a hierarchical control plane with a local and a meta control plane. The meta-control plane investigates the resources and the utilization of the local control plane to improve processing performance. In order to optimize data plane performance and remove the bottleneck of the central control, this mechanism supports the load balancing of the local control plane. The results have indicated that using the proposed meta control-based management mechanism, the loading of the control plane is reduced. Also, it improves the load balancing of SDN controllers and enhances the bandwidth utilization of the SDN environment relative to that obtained by means of single and multiple controllers without the meta control-based manager. But, it uses multiple controllers hence it suffers from high complexity. Also, energy consumption and execution time of the mechanisms have not been investigated.

Moreover, in the LTE networks, User Equipment (UEs) are commonly connected to a nearby cell (evolved Node B (eNB)), spatiotemporal variation in traffic requirements cause to load imbalance problem in the LTE networks. Because of the distributed nature of eNB operation in LTE Radio Access Network (RAN), conventional solutions to address load balancing issue might result in excessive overhead over X2 interface. Therefore, one of the challenging issues in the existing distributed LTE RAN is handling densely deployed cells. Rangisetti and Tamma [36] have presented a centralized Software Defined LTE RAN (SD-LTE-RAN) framework and a new QoS Aware Load Balance (QALB) algorithm. For making load balancing decisions, the algorithm considers loads of neighbor cells, QoS profiles of UEs and their estimated throughputs. The proposed method in comparison with other existing load balance algorithms (MinTHT and HLFB), maintains a remarkable degree of load balancing among cells and also improves entire network-wide GBR satisfaction and subsequently minimizes the total network overload. They have shown that the proposed QALB algorithm is able to preserve better QoS data rates for more than 80% of the cells in the network. In this paper, static UEs and mobility scenarios are also examined, these scenarios can reflect conditions in busy places such as airports, shopping malls, and railway stations. Not only in static scenarios, but also in mobile scenarios, QALB can minimize average network OLR in comparison to previous load balance algorithms. However, other QoS parameters such as delay and jitter have not been considered. Also, this method uses single controller hence it suffers from low scalability, low availability, and system bottleneck.

Finally, Lin *et al.* [37] have proposed a method to deal with Wi-Fi congestion in the SDN because of an unevenly distributed load among Access Points (APs). The conventional methods typically allow client stations know APs' load status and choose APs in a distributive manner. But, such a client-driven method lacks a global vision to make exact load balancing decisions and may result in frequent changes in the client-AP association. Their solution applies standardized OpenFlow protocol and SDN controller technology to establish the SDN controller and the APs into a two-tier architecture so that the controller can analyze the load balancing degree in the APs and choose which load level of the APs can accept association requests without referring to the controller. Experimental results have shown that proposed approach enhances the degree of Wi-Fi's load balancing and obtains a development of Wi-Fi's re-association time over generic centralized load balancing approaches with positive control. However, this approach uses single controller hence it suffers from low scalability, low availability, and system bottleneck. Also, they have not investigated more factors like user priorities, QoS limitations and traffic patterns of the associated devices in the load balancing decisions.

2) SUMMARY OF DETERMINISTIC APPROACHES

In this section, a side-by-side comparison of the selected deterministic techniques as well as their main advantages and disadvantages are shown in Table 3. Deterministic approaches have used both distributed controllers and centralized controller. Migration and re-routing techniques are some of the techniques which have been used in this category. Some of the advantages of these methods are: reducing overhead, improving the system throughput, improving the degree of load balancing and reducing average response time. Also, some of the disadvantages of these methods are unacceptable energy consumption and low availability. Also, these articles have not evaluated many factors.

B. NON-DETERMINISTIC APPROACHES

A non-deterministic approach is an algorithm that may have different behaviors on different runs for the same input. It often applied to acquire approximate solutions, when an exact solution is difficult or costly to acquire using a deterministic algorithm. In order to solve an NP-complete problem, a solution can be found by a nondeterministic algorithm in polynomial time. Considering that load balancing problem has the nature of NP-complete, it can be solved using a non-deterministic approach. In Section 1, the selected non-deterministic approaches are discussed and in Section 2, a summary of non-deterministic approaches are presented.

TABLE 3. Selected deterministic load balancing mechanisms and their properties.

Reference	Technique	Advantages	Disadvantages
Guo, et al. [17]	Load variance-based synchronization	• Reducing the synchronization overhead of controllers • Eliminating forwarding loops • Good performance • Without packet loss	• Latency has not evaluated • Energy consumption has not evaluated • The proposed algorithm has not evaluated in a real testbed
Yao, et al. [70]	Multicontroller load balancing approach (HybridFlow)	• Reducing the load jitter of multi-controller load • Relieving the working load on the super controller	• High complexity • Overhead of the algorithm and its throughput have not been investigated
Han and Ansari [43]	Virtualized green energy aware and latency aware user association	• Reducing the communication overheads between users and base stations • Reducing the on-grid power consumption	• Low scalability • Low availability • Increasing a small amount of average traffic delivery latency • Bottleneck • Throughput of the method has not been investigated
Duan, et al. [71]	Data offloading with load balancing in 5G HetNet	• Improving the degree of load balancing • Improving network stability • High throughput	• Low scalability • Low availability • Bottleneck • Packet loss rate and route delay have not been evaluated
Yong, et al. [72]	SDN-based dynamic load balance solution for cloud center (SDN-LB)	• Improving the system throughput • The load will not tilt over a long period of time	• Low scalability • Low availability • Bottleneck • Latency and utilization metrics have not been considered • Degree of load balancing metric has not been evaluated
Kang and Choo [53]	SDN-enhanced InterCloud Manager (S-ICM)	• Reducing average response time in the congested network • Suitable for a congested network	• Low scalability • Low availability • Bottleneck • Generates additional control messages into the entire network • Degree of load balancing metric has not been evaluated • Throughput has not been investigated
Raza, et al. [74]	Dynamically balance the load between multiple mobility anchors in SDN based proxy mobile IPv6 domain	• Improving in uplink and downlink traffic disruption periods	• Low scalability • Low availability • Bottleneck • Has not included the algorithms to perform load detection • Throughput and degree of load balancing metrics have not been considered
Song, et al. [47]	Lightweight load-balancing approach	• React more quickly when the network traffic is changed • Reducing the frequency of switch migration • High efficiently • Better throughput	• High complexity • Energy consumption metric has not considered • Latency has not been evaluated
Zhong, et al. [52]	Load balancing scheme based on server response (LBBSRT)	• Low cost • Good scalability • Easy to implement • High efficiently • Minimum response time • Medium resource utilization	• Low scalability • Low availability • Do not take into account the issue of energy saving in the load balancing • Bottleneck
Ma, et al. [75]	Meta controller-based manager mechanism	• High throughput • Reducing the loading of the control plane • Improving the load balancing of each SDN controller • Increasing the utilization of the bandwidth of the SDN environment	• High complexity • Energy consumption has not been investigated • Execution time of the mechanisms have not been evaluated
Rangisetti and Tamma [36]	QoS aware load balance (QALB)	• Maintaining better QoS data rates • Reducing average OLR • Better degree of load balancing • Minimizing the total network overload • Improving GBR	• Low scalability • Low availability • Without supporting jitter and delay • Bottleneck
Lin, et al. [37]	Two-tier dynamic load balancing	• Improving Wi-Fi's load balancing degree • Improving Wi-Fi's re-association time	• Low scalability • Low availability • Without supporting QoS constraints • Some factors such as traffic patterns of the associated devices and user priorities have not been evaluated • Bottleneck

1) OVERVIEW OF THE SELECTED NON-DETERMINISTIC APPROACHES

Chou *et al.* [19] have offered an OpenFlow-based load balancing system using the genetic algorithm. This system can efficiently distribute the data from clients to various servers based on load balancing strategies. Additionally, with the preconfigured flow table entries, each flow can be directed in advance. When the traffic burst or server loading unexpectedly increases, the proposed method can assist to balance the workload of server farms. The results have proved the

high efficiency of the proposed method in comparison to other methods including round-robin, random and load-based approaches. However, they have only used the arithmetic average for the coefficient of variation metric. Moreover, utilization and overhead of the algorithm have not been evaluated. Also, this method uses single controller hence availability and scalability are not achieved and the system has a bottleneck. Furthermore, since this method uses the genetic algorithm, it suffers from high computation time.

Additionally, the conventional traffic engineering techniques compute the optimal routing using a single traffic matrix. However, they are not able to address unexpected traffic adjustments. Accordingly, to deal with multiple feasible traffic scenarios, it is of interest to find a suitable routing structure. There are two main methods to reach load balancing for multiple traffic matrices; destination-based and explicit routing. It has been demonstrated that explicit routing does better than destination-based routing for multiple traffic matrices. On the other hand, explicit routing has high complexity and needs large Ternary Content Addressable Memory (TCAM) in the routers. Zhang *et al.* [42] has presented a method named hybrid routing to accomplish load balancing for multiple traffic matrices with high scalability and low complexity. The main idea of the proposed method is to complement destination-based routing with a small number of explicit routing forwarding entries to take benefit of both approaches. Hybrid routing substantially minimizes the number of forwarding entries in comparison to pure explicit routing. A heuristic algorithm has been implemented to attain the near-optimal hybrid routing configuration. The experimental results have shown that the hybrid routing obtains near-optimal load balancing compared to pure explicit routing. Specifically, a hybrid routing saves the TCAM resources in all practical networks. Whereas, latency, overhead, and utilization of the approach have not been considered.

On the other hand, existing data centers are not improved for cloud-based software services. Real data centers may suffer from lower throughput, as well as high latency and low QoS, because of the changeability of traffic load. Tu *et al.* [76] have introduced a programmable middlebox that can evenly distribute traffic to solve this problem. The middlebox is based on a Clos network, a multi-stage network which is proposed by Clos [77]. It designs and uses SDN to improve bandwidth utilization while ensuring QoS. The aim of the middlebox is to discover the optimal path for the traffic within the data center hence it uses a matrix called price matrix to depict the costs to transmit data from one to another server. A greedy algorithm is used to calculate the price matrix. The SDN controller of the middlebox gathers the information from switches and achieves load balancing by using the traffic distribution and server loads information. When it implemented in a data center, the middlebox can significantly enhance bandwidth utilization and minimizes latency. Since the middlebox does not rely on any specific characteristic of the data center, it can simply be applied in the existing data centers. However, they have not evaluated more QoS constraints. Also, the overhead of the middlebox and energy consumption of the approach have not been measured.

Moreover, the Internet of Vehicles (IoV) is considered as a typical application of the Internet of Things (IoT) in connection with the Intelligent Transportation System (ITS). Less mobility support, location awareness, and high processing latency still give rise to IoV suffering. To deal with the aforementioned problems, He, et al. [44] have integrated the fog computing with SDN. In order to apply the Software Defined Cloud/Fog Networking (SDCFN) architecture in the IoV, they have proposed a new SDN-based adapted constrained optimization particle swarm optimization algorithm which utilizes the reverse of the mutation particles flight and linear inertia weight to improve the efficiency of constrained optimization particle swarm optimization. The results have demonstrated that the proposed algorithm decreases the latency and improves the QoS in the SDCFN architecture. However, other QoS parameters such as security and capacity have not been evaluated. Also, energy consumption, load balancing degree, and utilization parameters have not been taken into consideration.

Current OpenFlow specification is not able to set the service rate of the queues inside OpenFlow devices. This lack does not let to apply most algorithms for the satisfaction of QoS requirements to new and established flows. Boero *et al.* [35] have proposed an alternative solution implemented over some modifications of Beacon (the popular SDN controller). The proposed solution uses real-time statistics from OpenFlow devices and Beacon will re-route flows on some queues to ensure the observance of deadline needs and/or an effective queue balancing in an OpenFlow SDN switch. Aiming at equalizing the traffic burden in each queue, three schemes have been proposed. Based on the greedy heuristic, one of these schemes is called multi-way. In the multi-way, at the beginning, all the flows are queued into q_0. The controller runs a system that sorts out the flows in decreasing order in compliance with the computed estimated rates. The established estimated rates order is analyzed and each flow is assigned to a queue with the lowest utilization to balance the load among the queues. The modification in the SDN controller will be the base for the design of a class of new re-routing algorithms which are able to ensure deadline constraints and queue balancing. They have not made any change to the OpenFlow specification, in addition to OpenFlow devices. Evaluation results have shown that multi-way schemes have a very satisfying behavior and better performance. However, they have not proposed any new primitive or modification of the OpenFlow standard. Also, the overhead of the approach and its execution time have not been investigated.

Furthermore, to facilitate the elasticity of the SDN controllers, the elastic scaling and the load balancing with effective switch migration is vital, however, it is still not easy to improve the migration efficiency. Wang *et al.* [40] have designed a switch migration scheme for the load balancing in the SDN controllers. They primarily have checked the

real-time controller load information gathered by the monitoring module and then decided whether to perform the switch migration. Later, to preserve the balance between the migration cost and the load balance rate, the migration efficiency model was built. Finally, based on a greedy method, they have designed an efficiency-aware migration algorithm to utilize the migration efficiency model and therefore, make the selection of migration actions feasible. According to the simulation results, the proposed method using switch migration makes the elasticity of SDN controllers possible and subsequently, enhances the migration efficiency. However, they have not implemented the method in a real large-scale wireless access network with more real-world traffic. Moreover, the latency of the method and its throughput have not been investigated.

Finally, in the SDN, it is usually required to regularly collect state/statistics of entire flows, which may cause too large overhead on control links. Xu *et al.* [48] have performed load-balanced routing using the traffic knowledge by carefully taking flow statistics collection to reduce the flow re-routing overhead. An important challenge is to achieve effective almost-optimal load-balanced routing with less overhead based on the quality of flow statistics collection. To deal with this challenge, a Partial Flow Statistics Collection (PFSC) problem is proposed, in which it is required to inquire statistics of flows from a subset of switches such that the flow recall ratio on every switch is at least a given value $\beta \in (0,1]$ while minimizing the number of queried switches. They have proved that the PFSC is an NP-Hard problem and have presented an algorithm relies on primal-dual with an approximation factor f/β in most situations, in which f is the maximum number of switches visited by each flow. To further reduce the overhead, they have designed an adaptive flow statistics collection mechanism, as a complementary scheme for PFSC, based on link load similarity measurement. Experimental and simulation results have shown that their methods can reduce the overhead in comparison to previous collection method while preserving a similar routing performance. However, the effects of the proposed method have not been evaluated when multiple controllers exist in the network. Moreover, the execution time of the approach and its latency, as well as its energy consumption, have not been considered.

2) SUMMARY OF NON-DETERMINISTIC APPROACHES

In this section, a side-by-side comparison of the selected nondeterministic techniques as well as their main advantages and disadvantages are shown in Table 4. Articles of non-deterministic approaches category have used methods including greedy, meta-heuristic and approximation. Genetic algorithm, multi-objective particle swarm, and particle swarm have been used in the articles of this category. Some of the advantages of these methods are: improving utilization, reducing latency and improving the degree of load balancing. Also, some of the disadvantages of this method are: high computationally time and unacceptable energy consumption. Additionally, these articles have not considered some of the metrics.

VI. RESULTS AND COMPARISON

This section will provide a review of the main advantages and disadvantages of deterministic and non-deterministic approaches and a comparative investigation of different load balancing metrics in SDN is also presented. In the previous sections, we have described some load balancing approaches in the SDN. These approaches are divided into two major distinct classes including deterministic and non-deterministic approaches.

An algorithm that always produces the same results for a particular input is called a deterministic approach. Based on our review and analysis in the sections IV and V, the most selected articles that used distributed controllers are belong to the category of deterministic approaches. Migration and re-routing techniques are some of the techniques that have been used in the selected articles. Most of the methods which have used the migration technique or have proposed a framework belong to the category of deterministic approaches. These methods can achieve the optimal solution, however, using a deterministic approach in the large-scale network will increase the runtime and latency. Many of the load balancing methods in the SDN have used deterministic approach.

A non-deterministic approach is an algorithm that may have various results on different runs for the same input. Articles of nondeterministic approaches category have used methods includes greedy, meta-heuristic and approximation. Genetic algorithm, multi-objective particle swarm, and particle swarm have been used in the articles of this category. Non-deterministic approaches usually have a lower complexity, but they suffer from scalability and availability metrics. These methods can find a good solution, but they may not achieve the optimal answer. Also, a non-deterministic approach requires a lot of memory and computation. Furthermore, we obtained different solutions when algorithm runs twice. Table 5 demonstrates main advantages and disadvantages of both deterministic and non-deterministic approaches.

In this study, selected articles have been evaluated for qualitative metrics. 19 metrics have detected the results of which are presented in Table 6. As shown in Fig. 6, researchers were focused on qualitative metrics as the degree of load balancing is 15%, throughput is 12%, utilization is 7%, execution time is 7% and latency is 10%. These results also show that degree of load balancing, throughput, utilization, execution time and latency metrics are the most crucial qualitative metrics that are used by researchers. Fig. 7 and Fig. 8 demonstrate the percentage of metrics for deterministic and nondeterministic techniques respectively. In deterministic approaches, the degree of load balancing and throughput metrics have the highest percentage and in non-deterministic approaches, latency and degree of load balancing metrics have the highest percentage.

TABLE 4. Selected non-deterministic load balancing techniques and their attributes.

Reference	Technique	Advantages	Disadvantages
Chou, et al. [19]	Genetic-based load balancing	• Better performance • Better degree of load balancing	• Low scalability • Low availability • Utilization has not been investigated • Overhead have not been evaluated • Bottleneck • High computationally time
Zhang, et al. [42]	Hybrid routing Destination-based routing with explicit routing	• Low complexity • Good scalability • Saving ternary content addressable memory resources • Achieving near-optimal load balancing • Reducing the number of forwarding entries	• Latency has not been considered • Overhead have not been investigated • Utilization has not been evaluated
Tu, et al. [76]	Programmable middlebox based on a Clos network	• Improving bandwidth utilization • Reducing latency • It can be deployed in current data centers	• Have not been evaluated more QoS constraints • Energy consumption has not been evaluated • Overhead of the middlebox has not been considered
He, et al. [44]	SDN-based modified constrained optimization particle swarm optimization	• Decreasing the latency • Improving the QoS	• Security and capacity have not evaluated • Energy consumption has not been considered • Utilization metric has not been evaluated • Degree of load balancing metric has not been evaluated
Boero, et al. [35]	Re-route flows on different queues	• Enabling to set the service rate of the queues inside OpenFlow devices • Without changing in new primitive or modification of the OpenFlow standard • Better Cumulative Frequency	• Execution time metric has not been considered • Overhead of the approach have not been measured
Wang, et al. [40]	Switch migration-based decision-making (SMDM)	• Improving migration efficiency • Enabling the elasticity of SDN controllers via switch migration • Low migration execution time • Low response time • Medium migration cost	• The method has not been implemented in a real large-scale network • Latency of the method and its throughput have not been measured
Xu, et al. [48]	Partial flow statistics collection (PFSC)	• Reducing the statistic collection overhead • Good performance	• The effects of this method have not been evaluated when multiple controllers • Execution time metric has not been evaluated • Latency and energy consumption metrics have not been investigated

TABLE 5. Main advantages and disadvantages of deterministic and non-deterministic approaches.

	Deterministic approaches	Nondeterministic approaches
Advantages	Better throughput Better degree of load balancing Better response time Better overhead	Better degree of load balancing Better utilization Better latency
Disadvantage	Unacceptable energy consumption Unacceptable latency Unacceptable availability Unacceptable packet loss	Unacceptable throughput Unacceptable energy consumption Unacceptable packet loss Unacceptable overhead

VII. OPEN ISSUE

This review shows that there are some important issues that have not been studied in the load distribution of SDN. Therefore, there are some open research problems that argued in this section.

In the some of the reviewed methods, factors such as traffic patterns and packet priorities have not been considered. Thus, one of the directions for future research can be applying these factors in the load balancing decisions.

It has been observed that there is not a particular mechanism to determine all QoS parameters for load balancing decisions. For example, some mechanisms consider throughput, scalability and response time while other parameters such as latency, packet loss, stability and etc. are ignored.

TABLE 6. Load balancing metrics in reviewed techniques.

Reference	Throughput	Peak load ratio	Utilization	Response time	Overhead	Root mean squared error	Packet loss rate	Percentage of matched deadline flow	Energy consumption	Migration cost	Forwarding entries	Execution time	Degree of load balancing	Guaranteed Bit Rate	Overload ratio	Average number of synchronizations per minute	Workload	Cumulative frequency	Latency
Song, et al. [47]	*				*							*					*		
Ma, et al. [75]	*		*																
Han and Ansari [43]									*										*
Kang and Choo [53]				*															
Yong, et al. [72]	*																		
Yao, et al. [70]													*				*		
Raza, et al. [74]																			*
Guo, et al. [17]					*	*	*									*			
Zhong, et al. [52]			*	*															
Rangisetti and Tamma [36]													*	*	*				
Duan, et al. [71]	*												*						
Lin, et al. [37]												*	*						
Wang, et al. [40]				*						*		*	*						
Xu, et al. [48]		*			*														
He, et al. [44]																			*
Boero, et al. [35]							*	*										*	
Zhang, et al. [42]	*										*								
Tu, et al. [76]			*																*
Chou, et al. [19]													*						

Therefore, load balancing decision making requires being extended to account more QoS parameters. Also, investigating how compliance with global QoS limitations may be very interesting.

Also, in the most of the studied techniques, researchers do not consider the challenge of energy saving and carbon emission. These factors are independently discussed and analyzed. In addition, they can be accepted to improve the popularity and effectiveness of existing load balancing mechanisms. Therefore, a load balancing technique based on carbon emission and energy consumption is extremely promising. Also, some of the reviewed methods have not been included the algorithm to perform load detection. Therefore, another direction for future works is a presentation of a new algorithm to carry out load detection.

Furthermore, failure management is one of the significant concepts that researchers have not been taken into account in existing load balancing mechanisms. Therefore, adding failure management to current methods is interesting in the future.

According to the heuristic-based mechanism, using new optimization methods can be very interesting for future works. Researchers can apply optimization methods such as honey bee swarm algorithm [78], lion optimization algorithm [79], whale optimization algorithm [80], gray wolf optimization algorithm [81], bat optimization algorithm [82] and etc. which can also be efficient to load balancing.

SDN permits efficient and flexible network management. But, flow tables capacity in SDN switches is limited so it is an obstacle to SDN development. Therefore, it is very interesting

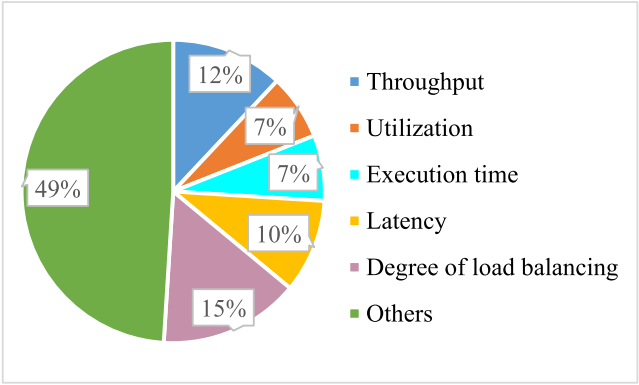

FIGURE 6. Percentage of load balancing metrics in reviewed techniques.

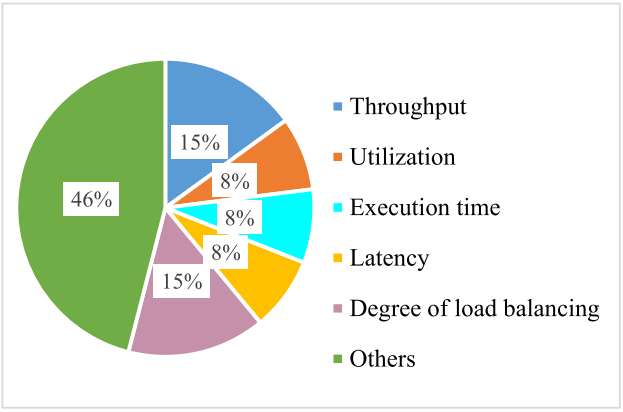

FIGURE 7. Percentage of load balancing metrics for deterministic approaches.

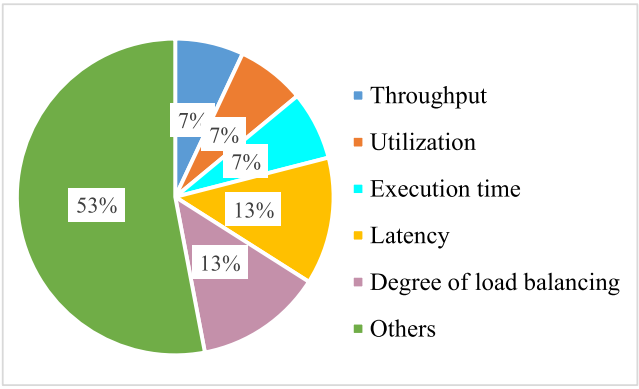

FIGURE 8. Percentage of load balancing metrics for nondeterministic approaches.

in the future to proposed routing methods that improve the efficiency of the flow tables.

Finally, applying an SDN based solution to provide load balancing in other similar networks such as peer-to-peer networks [83], mobile ad hoc networks [84], mobile cloud computing [85], machine-to-machine networks [86] is another research direction for future work.

VIII. CONCLUSION AND LIMITATION

This paper has provided a systematic review of load balancing mechanisms in the SDN. The SDN and load balancing were investigated and then the problem of load balancing in SDN was discussed. We described research methodology and selected 19 principal studies among the basic 136 papers from our search query. According to the SLR, the least articles in 2013 and the most articles have published in 2016. IEEE with 64 % of published articles present the highest published papers in conferences and journals. But, Sage with 2 %, have the smallest published articles among selected publishers. We classified 19 selected articles in two main categories that 12 of them are deterministic approaches and 7 of them are non-deterministic approaches. We also determined the used methods, advantages and disadvantages of the load balancing mechanisms. The SDN based solutions are used to provide load balancing in different networks such as LTE, cloud/fog, radio access, mobile and 5G networks. According to the results of previous sections, the SDN-based load balancing mechanisms apply a global view of the network hence these methods usually improve system performance compared to traditional load balancing approaches. In this paper, the challenges of these algorithms are investigated so that more effective load balancing techniques can be suggested in the future. The outcomes indicated that the most of the articles have not a specific mechanism to determine all QoS parameters for load balancing decisions. Also, the issue of energy saving and carbon emission are not discussed in the most of the reviewed techniques. Moreover, failure management has not gotten attention in existing load balancing mechanisms. Exclusively, the responses to the research questions summarized load balancing's main goal, existing challenges, mechanisms and open issues in SDN. We surely hope that the results of this research will assist researchers to expand more effective load balancing method in SDN.

We endeavored to provide the systematic and comprehensive review of load balancing mechanisms in SDN. However, it could have some constraints. Hence, future studies must consider the constraints of the current study as follow:

- ✓ Research Scope: The application of the load balancing in SDN has been covered in several sources such as thesis, editorial notes, technical reports, academic publications and web pages, etc. However, we considered only academic main international journals and conferences to achieve the best qualification. Also, we have ignored papers that published in national conferences and journals, as well as the papers not written in the English language, have removed. Our priority was to select journal articles but there were only fifteen journal articles in this field. Hence, we selected some conference papers.
- ✓ Study and publication bias: We selected eight online databases, based on prior review experiences. Whereas, the statistics indicate that this eight online database present the most relevant and reliable articles. But, selection of all related articles could not be guaranteed. There is a likelihood that some suitable articles were neglected throughout the processes discussed in Section IV.

- ✓ Study queries: We developed this article according to three questions that we defined but there is the probability of adding other questions.
- ✓ Classification: We categorized articles in deterministic and non-deterministic approaches but it can also be categorized differently.

REFERENCES

[1] S. Ortiz, "Software-defined networking: On the verge of a breakthrough?" *Computer*, vol. 46, no. 7, pp. 10–12, 2013.
[2] O. N. Foundation. (Jan. 2017). *Software-Defined Networking (SDN) Definition*. [Online]. Available: https://www.opennetworking.org/sdn-resources/sdn-definition
[3] A. Abdelaziz *et al.*, "Distributed controller clustering in software defined networks," *PLoS ONE*, vol. 12, no. 4, p. e0174715, 2017.
[4] "Software-defined networking: The new norm for networks," Open Netw. Found., ONF White Paper, 2012, pp. 2.6–6.1.
[5] N.-N. Dao, J. Kim, M. Park, and S. Cho, "Adaptive suspicious prevention for defending DoS attacks in SDN-based convergent networks," *PLoS ONE*, vol. 11, no. 8, p. e0160375, 2016.
[6] F. Pakzad, M. Portmann, W. L. Tan, and J. Indulska, "Efficient topology discovery in OpenFlow-based software defined networks," *Comput. Commun.*, vol. 77, pp. 52–61, Mar. 2016.
[7] A. Al-Najjar, S. Layeghy, and M. Portmann, "Pushing SDN to the end-host, network load balancing using OpenFlow," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Mar. 2016, pp. 1–6.
[8] S. Shenker, M. Casado, T. Koponen, and N. McKeown, "The future of networking, and the past of protocols," *Open Netw. Summit*, vol. 20, pp. 1–30, Oct. 2011.
[9] H. Xu, X.-Y. Li, L. Huang, H. Deng, H. Huang, and H. Wang, "Incremental deployment and throughput maximization routing for a hybrid SDN," *IEEE/ACM Trans. Netw.*, vol. 25, no. 3, pp. 1861–1875, 2017.
[10] T. Huang, F. R. Yu, C. Zhang, J. Liu, J. Zhang, and J. Liu, "A survey on large-scale software defined networking (SDN) testbeds: Approaches and challenges," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 891–917, 2nd Quart., 2016.
[11] W. Li, W. Meng, and L. F. Kwok, "A survey on OpenFlow-based software defined networks: Security challenges and countermeasures," *J. Netw. Comput. Appl.*, vol. 68, pp. 126–139, Jun. 2016.
[12] (Mar. 2015). *OpenFlow Switch Specification, 1.5.1*. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.1.pdf
[13] W. Lin and L. Zhang, "The load balancing research of SDN based on ant colony algorithm with job classification," in *Proc. 2nd Workshop Adv. Res. Technol. Ind. Appl.*, 2016, pp. 472–476.
[14] S. K. Askar, "Adaptive load balancing scheme for data center networks using software defined network," *Sci. J. Univ. Zakho*, vol. 4, no. 2, pp. 275–286, 2016.
[15] H. Long, Y. Shen, M. Guo, and F. Tang, "LABERIO: Dynamic load-balanced routing in OpenFlow-enabled networks," in *Proc. IEEE 27th Int. Conf. Adv. Inf. Netw. Appl. (AINA)*, Mar. 2013, pp. 290–297.
[16] T.-L. Lin, C.-H. Kuo, H.-Y. Chang, W.-K. Chang, and Y.-Y. Lin, "A parameterized wildcard method based on SDN for server load balancing," in *Proc. IEEE Int. Conf. Netw. Netw. Appl. (NaNA)*, Jul. 2016, pp. 383–386.
[17] Z. Guo *et al.*, "Improving the performance of load balancing in software-defined networks through load variance-based synchronization," *Comput. Netw.*, vol. 68, pp. 95–109, Aug. 2014.
[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. 2016.
[19] L.-D. Chou, Y.-T. Yang, Y.-M. Hong, J.-K. Hu, and B. Jean, "A genetic-based load balancing algorithm in OpenFlow network," in *Advanced Technologies, Embedded and Multimedia for Human-Centric Computing*. Dordrecht, The Netherlands: Springer, 2014, pp. 411–417.
[20] K. Benzekki, A. El Fergougui, and A. E. Elalaoui, "Software-defined networking (SDN): A survey," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5803–5833, 2016.
[21] O. N. Foundation. (Jun. 2014). *SDN Architecture*. [Online]. Available: https://www.opennetworking.org/images/stories/downloads/sdn-resources/technical-reports/TR_SDN_ARCH_1.0_06062014.pdf
[22] F. Hu, Q. Hao, and K. Bao, "A survey on software-defined network and OpenFlow: From concept to implementation," *IEEE Commun. Surv. Tuts.*, vol. 16, no. 4, pp. 2181–2206, 4th Quart., 2014.
[23] S. J. Vaughan-Nichols, "OpenFlow: The next generation of the network?" *Computer*, vol. 44, no. 8, pp. 13–15, 2011.
[24] W. Zhou, S. Yang, J. Fang, X. Niu, and H. Song, "VMCTune: A load balancing scheme for virtual machine cluster using dynamic resource allocation," in *Proc. IEEE 9th Int. Conf. Grid Cooperat. Comput. (GCC)*, Nov. 2010, pp. 81–86.
[25] M. L. Chin, C. E. Tan, and M. I. Bandan, "Efficient load balancing for bursty demand in Web based application services via domain name services," in *Proc. IEEE 8th Asia–Pacific Symp. Inf. Telecommun. Technol. (APSITT)*, Jun. 2010, pp. 1–4.
[26] M. A. Salman, C. Bertelle, and E. Sanlaville, "The behavior of load balancing strategies with regard to the network structure in distributed computing systems," in *Proc. IEEE 10th Int. Conf. Signal-Image Technol. Internet-Based Syst. (SITIS)*, Nov. 2014, pp. 432–439.
[27] A. S. Milani and N. J. Navimipour, "Load balancing mechanisms and techniques in the cloud environments: Systematic literature review and future trends," *J. Netw. Comput. Appl.*, vol. 71, pp. 86–98, Aug. 2016.
[28] S.-L. Chen, Y.-Y. Chen, and S.-H. Kuo, "CLB: A novel load balancing architecture and algorithm for cloud services," *Comput. Elect. Eng.*, vol. 58, pp. 154–160, Feb. 2017.
[29] A. M. Alakeel, "A guide to dynamic load balancing in distributed computer systems," *Int. J. Comput. Sci. Inf. Secur.*, vol. 10, no. 6, pp. 153–160, 2010.
[30] A. Tootoonchian and Y. Ganjali, "Hyperflow: A distributed control plane for OpenFlow," in *Proc. Internet Netw. Manage. Conf. Res. Enterprise Netw.*, 2010, p. 3.
[31] P. Lin, J. Bi, and H. Hu, "ASIC: An architecture for scalable intra-domain control in OpenFlow," in *Proc. ACM 7th Int. Conf. Future Internet Technol.*, 2012, pp. 21–26.
[32] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On scalability of software-defined networking," *IEEE Commun. Mag.*, vol. 51, no. 2, pp. 136–141, Feb. 2013.
[33] X. Huang, S. Bian, Z. Shao, and H. Xu, "Dynamic switch-controller association and control devolution for SDN systems," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[34] G. Cheng, H. Chen, H. Hu, and Z. Wang, "Toward a scalable SDN control mechanism via switch migration," *China Commun.*, vol. 14, no. 1, pp. 111–123, 2017.
[35] L. Boero, M. Cello, C. Garibotto, M. Marchese, and M. Mongelli, "BeaQoS: Load balancing and deadline management of queues in an OpenFlow SDN switch," *Comput. Netw.*, vol. 106, pp. 161–170, Sep. 2016.
[36] A. K. Rangisetti and B. R. Tamma, "QoS Aware load balance in software defined LTE networks," *Comput. Commun.*, vol. 97, pp. 52–71, Jan. 2017.
[37] Y.-D. Lin, C. C. Wang, Y.-J. Lu, Y.-C. Lai, and H.-C. Yang, "Two-tier dynamic load balancing in SDN-enabled Wi-Fi networks," *Wireless Netw.*, vol. 23, pp. 1–13, Apr. 2017.
[38] Y. Hu, T. Luo, N. C. Beaulieu, and W. Wang, "An initial load-based green software defined network," *Appl. Sci.*, vol. 7, no. 5, p. 459, 2017.
[39] Y. Carlinet and N. Perrot, "Energy-efficient load balancing in a SDN-based Data-Center network," in *Proc. IEEE 17th Int. Telecommun. Netw. Strategy Planning Symp. (Networks)*, Sep. 2016, pp. 138–143.
[40] C. Wang, B. Hu, S. Chen, D. Li, and B. Liu, "A switch migration-based decision-making scheme for balancing load in SDN," *IEEE Access*, vol. 5, pp. 4537–4544, 2017.
[41] R. Zhu, H. Wang, Y. Gao, S. Yi, and F. Zhu, "Energy saving and load balancing for SDN based on multi-objective particle swarm optimization," in *Proc. Int. Conf. Algorithms Archit. Parallel Process.*, 2015, pp. 176–189.
[42] J. Zhang, K. Xi, M. Luo, and H. J. Chao, "Load balancing for multiple traffic matrices using SDN hybrid routing," in *Proc. IEEE 15th Int. Conf. High Perform. Switching Routing (HPSR)*, Jul. 2014, pp. 44–49.
[43] T. Han and N. Ansari, "A traffic load balancing framework for software-defined radio access networks powered by hybrid energy sources," *IEEE/ACM Trans. Netw.*, vol. 24, no. 2, pp. 1038–1051, Apr. 2016.
[44] X. He, Z. Ren, C. Shi, and J. Fang, "A novel load balancing strategy of software-defined cloud/fog networking in the Internet of Vehicles," *China Commun.*, vol. 13, pp. 140–149, Nov. 2016.
[45] C. Liang, R. Kawashima, and H. Matsuo, "Scalable and crash-tolerant load balancing based on switch migration for multiple open flow controllers," in *Proc. IEEE 2nd Int. Symp. Comput. Netw. (CANDAR)*, Dec. 2014, pp. 171–177.

[46] N. T. Hai and D.-S. Kim, "Efficient load balancing for multi-controller in SDN-based mission-critical networks," in *Proc. IEEE 14th Int. Conf. Ind. Inform. (INDIN)*, Jul. 2016, pp. 420–425.
[47] P. Song, Y. Liu, T. Liu, and D. Qian, "Flow Stealer: Lightweight load balancing by stealing flows in distributed SDN controllers," *Sci. China Inf. Sci.*, vol. 60, no. 3, p. 032202, 2017.
[48] H. Xu, X.-Y. Li, L. Huang, Y. Du, and Z. Liu, "Partial flow statistics collection for load-balanced routing in software defined networks," *Comput. Netw.*, vol. 122, pp. 43–55, Jul. 2017.
[49] F. S. Fizi and S. Askar, "A novel load balancing algorithm for software defined network based datacenters," in *Proc. IEEE Int. Conf. Broadband Commun. Next Generat. Netw. Multimedia Appl. (CoBCom)*, Sep. 2016, pp. 1–6.
[50] D. Adami, S. Giordano, M. Pagano, and N. Santinelli, "Class-based traffic recovery with load balancing in software-defined networks," in *Proc. IEEE Globecom Workshops (GC Wkshps)*, Dec. 2014, pp. 161–165.
[51] S. Sharma, S. Singh, and M. Sharma, "Performance analysis of load balancing algorithms," *World Acad. Sci., Eng. Technol.*, vol. 38, no. 3, pp. 269–272, 2008.
[52] H. Zhong, Y. Fang, and J. Cui, "LBBSRT: An efficient SDN load balancing scheme based on server response time," *Future Generat. Comput. Syst.*, vol. 68, pp. 183–190, Mar. 2017.
[53] B. Kang and H. Choo, "An SDN-enhanced load-balancing technique in the cloud system," *J. Supercomput.*, vol. 3, pp. 1–24, Dec. 2016.
[54] D. Levin, A. Wundsam, B. Heller, N. Handigol, and A. Feldmann, "Logically centralized?: State distribution trade-offs in software defined networks," in *Proc. ACM 1st Workshop Hot Topics Softw. Defined Netw.*, 2012, pp. 1–6.
[55] E. Y. Daraghmi and S.-M. Yuan, "A small world based overlay network for improving dynamic load-balancing," *J. Syst. Softw.*, vol. 107, pp. 187–203, Sep. 2015.
[56] R. Subramanian and T. Manoranjitham, "Dynamic scheduling for traffic management and load balancing using SDN," *Int. Sci. Press*, vol. 9, no. 2, pp. 919–925, 2016.
[57] D. W. Deepika and N. Kumar, "Performance analysis of load balancing algorithms in distributed system," *Adv. Electron. Electr. Eng.*, vol. 4, no. 1, pp. 59–66, 2014.
[58] D. Li, S. Wang, K. Zhu, and S. Xia, "A survey of network update in SDN," *Frontiers Comput. Sci.*, vol. 11, no. 1, pp. 4–12, 2017.
[59] C. Trois, M. D. Del Fabro, L. C. E. de Bona, and M. Martinello, "A survey on SDN programming languages: Toward a taxonomy," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 4, pp. 2687–2712, 4th Quart., 2016.
[60] S. Schmid, K. G. Larsen, and B. Xue, "WNetKAT: A weighted SDN programming and verification language," in *Proc. 20th Int. Conf. Principles Distrib. Syst. (OPODIS)*, 2016, pp. 1–18.
[61] S. Rowshanrad, S. Namvarasl, V. Abdi, M. Hajizadeh, and M. Keshtgary, "A survey on SDN, the future of networking," *J. Adv. Comput. Sci. Technol.*, vol. 3, no. 2, p. 232, 2014.
[62] Y. Jarraya, T. Madi, and M. Debbabi, "A survey and a layered taxonomy of software-defined networking," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1955–1980, 4th Quart., 2014.
[63] M. Karakus and A. Durresi, "Quality of service (QoS) in software defined networking (SDN): A survey," *J. Netw. Comput. Appl.*, vol. 80, pp. 200–218, Feb. 2016.
[64] B. Kitchenham, "Procedures for performing systematic reviews," Dept. Comput. Sci., Keele Univ., Keele, U.K., Tech. Rep. TR/SE-0401, and NICTA Tech. Rep. 0400011T.1, Eversleigh, NSW, Australia, 2004, vol. 33, pp. 1–26.
[65] E. Kupiainen, M. V. Mäntylä, and J. Itkonen, "Using metrics in agile and lean software development—A systematic literature review of industrial studies," *Inf. Softw. Technol.*, vol. 62, pp. 143–163, Jun. 2015.
[66] Y. Charband and N. J. Navimipour, "Online knowledge sharing mechanisms: A systematic review of the state of the art literature and recommendations for future research," *Inf. Syst. Frontiers*, vol. 18, no. 6, pp. 1131–1151, 2016.
[67] N. J. Navimipour and Y. Charband, "Knowledge sharing mechanisms and techniques in project teams: Literature review, classification, and current trends," *Comput. Hum. Behav.*, vol. 62, pp. 730–742, Sep. 2016.
[68] W. Reim, V. Parida, and D. Örtqvist, "Product–service systems (PSS) business models and tactics—A systematic literature review," *J. Cleaner Prod.*, vol. 97, pp. 61–75, Jun. 2015.
[69] B. Kitchenham, O. P. Brereton, D. Budgen, M. Turner, J. Bailey, and S. Linkman, "Systematic literature reviews in software engineering—A systematic literature review," *Inf. Softw. Technol.*, vol. 51, no. 1, pp. 7–15, 2009.
[70] H. Yao, C. Qiu, C. Zhao, and L. Shi, "A multicontroller load balancing approach in software-defined wireless networks," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 10, p. 454159, 2015.
[71] X. Duan, A. M. Akhtar, and X. Wang, "Software-defined networking-based resource management: Data offloading with load balancing in 5G HetNet," *EURASIP J. Wireless Commun. Netw.*, vol. 2015, p. 181, Dec. 2015.
[72] W. Yong, T. Xiaoling, H. Qian, and K. Yuwen, "A dynamic load balancing method of cloud-center based on SDN," *China Commun.*, vol. 13, no. 2, pp. 130–137, Feb. 2016.
[73] N. Handigol, S. Seetharaman, M. Flajslik, N. McKeown, and R. Johari, "Plug-n-Serve: Load-balancing Web traffic using OpenFlow," *ACM SIGCOMM Demo*, vol. 4, no. 5, p. 6, Aug. 2009.
[74] S. M. Raza, D. Park, Y. Park, K. Lee, and H. Choo, "Dynamic load balancing of local mobility anchors in software defined networking based proxy mobile IPv6," in *Proc. ACM 10th Int. Conf. Ubiquitous Inf. Manage. Commun.*, 2016, p. 106.
[75] Y.-W. Ma, J.-L. Chen, Y.-H. Tsai, K.-H. Cheng, and W.-C. Hung, "Load-balancing multiple controllers mechanism for software-defined networking," *Wireless Pers. Commun.*, vol. 94, no. 4, pp. 3549–3574, 2017.
[76] R. Tu, X. Wang, J. Zhao, Y. Yang, L. Shi, and T. Wolf, "Design of a load-balancing middlebox based on SDN for data centers," in *Proc. IEEE Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr./May 2015, pp. 480–485.
[77] C. Clos, "A study of non-blocking switching networks," *Bell Labs Tech. J.*, vol. 32, no. 2, pp. 406–424, 1953.
[78] D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Dept. Eng. Faculty, Comput. Eng., Erciyes Univ., Kayseri, Turkey, Tech. Rep. TR06, 2005.
[79] M. Yazdani and F. Jolai, "Lion optimization algorithm (LOA): A nature-inspired metaheuristic algorithm," *J. Comput. Des. Eng.*, vol. 3, no. 1, pp. 24–36, 2016.
[80] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Adv. Eng. Softw.*, vol. 95, pp. 51–67, May 2016.
[81] S. Mirjalili, S. M. Mirjalili, and A. Lewis, "Grey wolf optimizer," *Adv. Eng. Softw.*, vol. 69, pp. 46–61, Mar. 2014.
[82] X.-S. Yang and A. H. Gandomi, "Bat algorithm: A novel approach for global engineering optimization," *Eng. Comput.*, vol. 29, no. 5, pp. 464–483, 2012.
[83] N. J. Navimipour and F. S. Milani, "A comprehensive study of the resource discovery techniques in Peer-to-Peer networks," *Peer-to-Peer Netw. Appl.*, vol. 8, no. 3, pp. 474–492, 2015.
[84] R. Geng, Z. Ning, and N. Ye, "A load-balancing and coding-aware multicast protocol for mobile *ad hoc* networks," *Int. J. Commun. Syst.*, vol. 29, no. 17, pp. 2457–2470, 2016.
[85] N. Fernando, S. W. Loke, and W. Rahayu, "Mobile cloud computing: A survey," *Future Generat. Comput. Syst.*, vol. 29, no. 1, pp. 84–106, 2013.
[86] L. Liu, T. Zhang, and J. Zhang, "DAG based multipath routing algorithm for load balancing in machine-to-machine networks," *Int. J. Distrib. Sensor Netw.*, vol. 10, no. 1, p. 457962, 2013.

ALI AKBAR NEGHABI received the B.S. degree in computer engineering, specialized in software engineering, from the Sadjad University of Technology, Mashhad, Iran, in 2002, and the M.Sc. degree in computer engineering, specialized in software engineering, from the Arak Branch, Islamic Azad University, Tehran, Iran, in 2005. He is currently pursuing the Ph.D. degree with the Department of Computer Engineering, Science and Research Branch, Islamic Azad University. His research interests include cloud systems, social networks, and recommender systems.

NIMA JAFARI NAVIMIPOUR received the B.S. degree in computer engineering, specialized in software engineering, and the M.S. degree in computer engineering, specialized in computer architecture, from the Tabriz Branch, Islamic Azad University, Tabriz, Iran, in 2007 and 2009, respectively, and the Ph.D. degree in computer engineering, specialized in computer architecture, from the Science and Research Branch, Islamic Azad University, in 2014. He is currently an Assistance Professor with the Department of Computer Engineering, Tabriz Branch, Islamic Azad University. His research interests include SDN, cloud computing, grid systems, computational intelligence, evolutionary computing, and wireless networks.

MEHDI HOSSEINZADEH received the B.E. degree in computer hardware engineering from the Dezful Branch, Islamic Azad University (IAU), Tehran, Iran, in 2003, and the M.Sc. and Ph.D. degrees in computer system architecture from the Science and Research Branch, IAU, in 2005 and 2008, respectively. He is currently an Associate Professor with the Iran University of Medical Sciences, Tehran. His research interests include SDN, information technology, data mining, big data analytics, E-commerce, E-marketing, and social networks.

ALI REZAEE received the B.S. and M.S. degrees in software engineering in 2005 and 2008, respectively, and the Ph.D. degree in software systems from IAU in 2012. He is currently an Assistant Professor with the Science and Research Branch, Islamic Azad University (IAU). He is also the Chair of the Distributed Systems Laboratory, and the Formal Verification Laboratory, IAU. He is also lead system architect at Open Farm Technologies. His main research interests include distributed systems, IoT, formal verification, operating systems, and software architecture.