Computer Networks 91 (2015) 528–547

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

Survey Paper

# A survey on data center networking for cloud computing

Bin Wang [a,*], Zhengwei Qi [a], Ruhui Ma [a], Haibing Guan [a], Athanasios V. Vasilakos [b]

[a] School of Software, Shanghai Jiao Tong University, Shanghai, China
[b] Department of Computer Science, Electrical and Space Engineering, Lulea University of Technology, Sweden

ARTICLE INFO

*Article history:*
Received 30 October 2014
Revised 2 June 2015
Accepted 27 August 2015
Available online 10 September 2015

*Keywords:*
Cloud computing
Data center networking
Virtualization
Software defined networking

ABSTRACT

Data Center Networks (DCNs) are an essential infrastructure that impact the success of cloud computing. A scalable and efficient data center is crucial in both the construction and operation of stable cloud services. In recent years, the growing importance of data center networking has drawn much attention to related issues including connective simplification and service stability. However, existing DCNs lack the necessary agility for multi-tenant demands in the cloud, creating poor responsiveness and limited scalability. In this paper, we present an overview of data center networks for cloud computing and evaluate construction prototypes based on these issues. We provide, specifically, detailed descriptions of several important aspects: the physical architecture, virtualized infrastructure, and DCN routing. Each section of this work discusses and evaluates resolution approaches, and presents the use cases for cloud computing service. In our attempt to build insight relevant to future research, we also present some open research issues. Based on experience gained in both research and industrial trials, the future of data center networking must include careful consideration of the interactions between the important aspects mentioned above.



## 1. Introduction

Cloud services have increased rapidly in number and scale across many application areas. Recent reports showed that the cloud service market reached a size of $ 148.9 billion by 2014 [1]. As of today, a simple web search request may touch 1000+ servers, while a large computing request sometimes consumes thousands of machines [2]. Cloud computing requires the underlying network to be fast, carry large amounts of traffic, and be scalable. Building such complex environments using traditional networks is time-consuming and expensive.

Data center is a set of servers, storage and network devices, power systems, cooling systems, etc [3]. Data centers are intended for large-scale service applications such as online businesses, Smart Grid [3,4] and scientific computation [2]. Data Center Network (DCN) includes data center and provides the connections to the data center, which is described by its network topology, routing/switching equipment, and the protocols it uses [3,4]. DCN offers many features to help organize cloud computing for the following reasons:

- DCN permits the connection of thousands of data center servers in an efficient way, so that cloud computing could simply expand its service by following the DCN topology.
- DCN offers traffic reliability and efficiency to massive machine-to-machine communications in which activities from cloud computing emerge as the workloads distributed on data center servers.
- DCN supports various virtualization techniques that help DCN to create Virtual Machine (VM) [5], virtual network, and virtual function. DCN should possess the scalability to provide isolation and migration to massive numbers of virtual instances.

* Corresponding author. Tel.: +8618516199503.
*E-mail addresses:* binqbu2002@sjtu.edu.cn (B. Wang), qizhwei@sjtu.edu.cn (Z. Qi), ruhuima@sjtu.edu.cn (R. Ma), hbguan@sjtu.edu.cn (H. Guan), th.vasilakos@gmail.com (A.V. Vasilakos).

http://dx.doi.org/10.1016/j.comnet.2015.08.040
1389-1286/

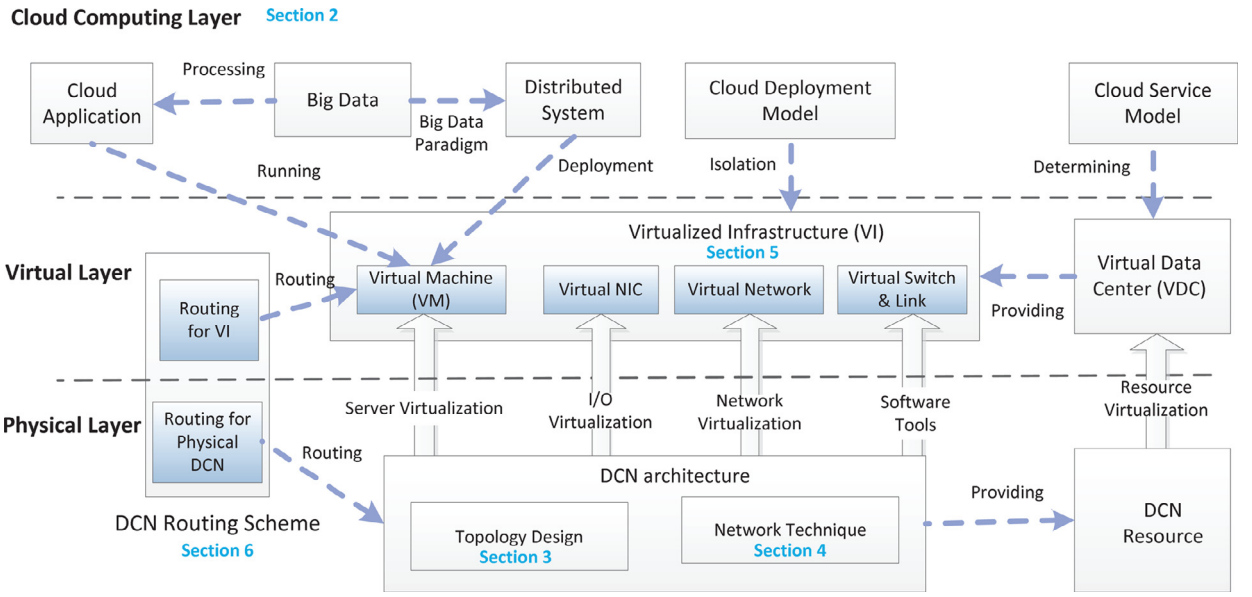


**Fig. 1.** Cloud computing network architecture.

- Existing research on DCNs has generated solutions for many use cases such as green computing, data center backup, some of which can also address the challenges of cloud service.

### 1.1. Cloud computing and DCN

Fig. 1 depicts cloud computing architecture that illustrates the separation of cloud computing layer, virtual layer and physical layer. The physical layer is mainly responsible for offering hardware resources such as CPU, memory, and bandwidth, and for enforcing routing and traffic management in a real-world DCN. Activities from the other two layers become workloads, where the high-level activities involve information being tunneled and transmitted by various physical agents, e.g., server, switch, or middlebox. Virtual layer offers virtual instances such as VM and virtual network to the cloud users. In this layer, a virtual resource is allocated according to various policies, which serve the cloud service model. The last layer is the cloud computing layer, which arranges all the cloud activities on top of the virtual layer. In this paper, cloud activities mainly include cloud service model, cloud deployment model, big data [6], and cloud distributed system. All these activities make great demands on the other layers offered by DCN in terms of traffic efficiency, on-demand allocation, and reliability.

### 1.2. DCN performance in cloud computing service

We will briefly discuss the performance that remains for existing DCN techniques. The key findings of this paper are:

- Most DCN architectures are not optimally designed for use with cloud computing. For example, the tree-based DCNs, such as Fat-tree or Three-tier [7–9], are the most deployed architectures in the enterprise cloud. However, these architectures lack scalability in terms of load balancing and rate control, which make them poor at managing traffic from thousands of servers and VMs. On the other hand, the newer architectures for cloud computing are not effective at addressing or topology self-learning, and as such are currently not used as the DCN architectures for cloud computing.
- Scalability is the most constraining problem for DCN solutions that support cloud computing. In particular, it constrains both centralized and distributed approaches to routing, control plane management, resource allocation, etc. We believe that this problem could be solved with the collaboration of different network elements. For example, it is impossible for a centralized load balancing approach to collect and control every switch in a large-scale DCN, but a solution could reduce computational workloads in the centralized controller if some flows are decided locally in a commodity switch.
- DCN architectures are divergent in topology. This makes it difficult to manage general routing, virtualization, and load balancing in the higher layers. For example, many load balancers are fast at rerouting local flow [10]. However, since they are not aware of the switch locations or loads elsewhere, some DCN architectures with redundant path design will undermine the load balancing performance with a forwarding loop. On the other hand, DCN-specific solutions lack interfaces for Internet access. For example, some DCNs customize their routing protocols in a way that might not provide the interface for the Border Gateway Protocol (BGP) [11] at layer 3.

### 1.3. Objectives and organization

This paper presents a comprehensive overview of the existing data center networks for cloud computing. The goal of this paper is to understand the challenges and research

opportunities for DCNs that support cloud service, including physical DCN architecture, Virtualized Infrastructure (VI), and DCN routing. Fig. 1 also shows an outline of this paper, which organizes and links different sections as follows: Section 2 gives DCN requirements and challenges in use of DCN for cloud service, all of which are depicted at cloud computing layer in Fig. 1. Sections 3 and 4 overview existing DCN architectures and discuss the current research issues for cloud service provision. Section 5 categorizes methods of network virtualization and discusses VI, which appears at virtual layer in Fig. 1. Besides, the resource allocation for VI would be described as a Virtual Data Center (VDC) [12] in this section. Section 6 elaborates upon the routing and forwarding schemes used in DCNs and covers two layers in Fig. 1. Finally, Section 7 provides a conclusion.

### 1.4. Related work

Many previous studies have explored the functions of DCNs. Chen et al. summarized the routing function in DCNs after thoroughly investigating different routing architectures and classifying them as server-centric, switch-centric or hybrid [2]. Similarly, a survey conducted by Abts and Felderman explored legacy data centers in terms of traffic, routing, and energy saving [3]. In [4], Chen et al. classified existing wireless and optical DCN infrastructures while introducing their novel data center network model. In [13], Bari et al. discussed the main virtualization infrastructures in routing, traffic, and resource management. In their discussion of energy efficiency and green data centers, Bilal et al. conducted a survey that details physical data center architectures, traffic management techniques, and monitors [8]. In this paper, we provide further detailed classification and a conclusive overview of existing cloud DCNs. Our survey details pertinent DCN topics such as routing and virtualization that can be effectively deployed as-is for cloud services.

## 2. DCN requirements and challenges for cloud activities

Data centers provide both physical and virtual infrastructures to a cloud computing system. DCNs are responsible for the QoS of connectivity and communication in the cloud computing system. Activities in the cloud have specific networking requirements that facilitate and challenge the existing DCN design. In this section, we describe the main requirements and challenges of using DCN for cloud activities.

### 2.1. DCN for cloud service model

An essential use case of DCNs for cloud computing is that they provide the infrastructure for cloud computing service models. The US National Institute of Standards and Technology (NIST) [14] defines various service models of cloud computing. The three most famous service models are Infrastructure as a Service (IaaS) [15], Platform as a Service (PaaS) [16], and Software as a Service (SaaS) [16], which have already been deployed in today's enterprise data centers.

- **DCN for IaaS model:** To serve the IaaS model, DCN should first virtualize adequate infrastructure resources for cloud use. Second, DCNs should map all IaaS VMs to DCN servers so that infrastructure resources on each server could be fully utilized. Finally, DCNs are responsible for the maintenance, repair, and migration of virtual infrastructure resources on each VM.
- **DCN for PaaS model:** To serve this model, libraries and tools from VMs are abstracted as virtual instances of applications on DCN servers. The DCN also performs maintenance, load balancing, and scale-up of the platform.
- **DCN for SaaS model:** This model does not require each end-user to manually download, install, configure, run, or use the software applications on their own computing environments, as all of these tasks are performed by the cloud computing system. Since each DCN server runs multiple compute-intensive tasks for massive user-space services, SaaS requires a DCN to be scalable in order to support massive parallel computing applications such as big data system [6].

There are two challenges to the DCN for cloud service model. Firstly, since cloud service model provides on-demand flexibility in both resource and service managements, each cloud utility must be able to start and stop allocating resources or services on the fly. This causes a time-varying demand that requires the DCN to dynamically allocate and utilize hardware resources and services. On the other hand, most cloud service models advocate centralized management in the DCN in order to schedule available resources or services across the whole network [3]. Since today's DCNs usually expand the scale for cloud computing, the centralized management for on-demand allocation lowers the efficiency of large-scale cloud service.

Secondly, since a cloud service model serves the commercial goals of the Cloud Service Provider (CSP), there are many systems deployed in the cloud that are responsible for computing service pricing. However, the existing DCN traffic problems typically affect the tracing of the price calculation. For example, IaaS cloud services charge each infrastructure resource by time [17]; thus, a poor DCN environment where congestion occurs frequently will generate time delays that waste resources and money.

### 2.2. DCN for cloud deployment model

Cloud computing offers different deployment models for the implementation of cloud services. There are three common models for the enterprise cloud: public, private, and hybrid [14]. The requirement details are explained as follows.

- **DCN for public cloud:** In this deployment model, all public users pay per use of services according to Service Level Aggrement (SLA) [18]. There is not a strict requirement to coordinate security, load balancing, and routing in the DCN, but the specific QoS relies on the cloud service model. Since customer requirements of cloud services usually vary, the DCN has to ensure that it can be flexible in its service delivery.
- **DCN for private cloud:** This model offers the highest degree of control over performance, reliability, and security to the cloud user. The DCN should create a virtual private network to install the private cloud model for each user. Such a network should be highly reliable and isolate user disk space and permissions.

- **DCN for hybrid cloud:** This model is a combination of a public and private cloud. In a hybrid model, users keep sensitive data in the private cloud while outsourcing non-business critical information and processing to the public cloud. The goal of a DCN for a hybrid cloud is to determine the optimal boundary between the public and private cloud components. For example, the DCN should prevent data loss when a user outsources non-sensitive data to the public cloud, while maintaining the reliability of each private cloud.

Security and network isolation are the essential features that govern different deployment models of DCN technique. However, the existing network virtualization technique has limitation since it assumes the DCN belongs to the well-known structures such as Three-tier [7]. In practice, it is hard to enumerate all possible structures for DCNs. Additionally, since many network virtualization techniques isolate different virtual networks by segmenting packet header bits, they require modification and configuration to each router and switch in the DCN (details in Section 5).

### 2.3. DCN and big data

An essential feature of a cloud-based platform is facilitating the rapid processing of large data sets in cloud applications, as end users increasingly demand urgent processing of emerging or rapidly evolving data sets [19]. With the expansion of big data needs, providers have increasingly supported big data using cloud-based platforms. To deploy big data applications on a cloud DCN, volume, velocity, and variety are key challenges. Volume refers to the amount of data, variety refers to the number of data types [6], and velocity refers to the speed of data processing. The details of these challenges are explained as follows.

- **Big data volume problems:** A cloud DCN should provide the virtual network architecture that acquires and stores large volumes of data. In this respect, the VM used for a big data node should have a large capacity for data acquisition, processing, organization, and analysis. This demands the cloud DCN be able to place more CPUs, memory, and offline storage resources for these VMs, which incurs challenges to resource computing and scheduling in the DCN.
- **Big data variety problems:** Since a big data system can select different workflows to process different data structures, the virtual network should support this via dispatching to parallel compute cores. This feature is a challenge to existing network virtualization techniques in their configuration of the network topology in the DCNs routers. For example, a VM used for a big data node could be migrated to another server for a different workflow, which incurs a migration overhead on the networks load.
- **Big data velocity problems:**To support high velocity data throughput, a cloud DCN should improve the QoS of virtual networks. One challenge is that the virtual links in a big data system should have large capacities. That requires the VMs used for big data nodes on each server to have larger bandwidth in the network. This causes a waste of physical bandwidth when idle VMs do not share the bandwidth.

### 2.4. DCN for cloud distributed system

Cloud Distributed Systems (CDS) are used for big data and cloud computing paradigms because they efficiently partition and aggregate data based on various key-value algorithms. Many CDSs have emerged as popular subset tools of big data systems, e.g., MapReduce [20] and Spark [19] for Hadoop or DryadLINQ [6] for Microsoft clusters [21]. At the network level, the task of a CDS is to produce workflows that process all the cloud data. A workflow will assign a number of worker and aggregator nodes in the DCN. In general, data originating from a cloud application could be processed by three kinds of workflows, which have the following DCN requirements:

- **Partition-aggregate workflow:**In this workflow, an aggregator node divides computation across multiple worker nodes. Worker nodes perform the computation in parallel and send results back to the aggregator. The aggregator combines these results to provide a complete response. Applications that are processed using a partition-aggregate workflow include web search and data ranking. To set up a partition-aggregate workflow, a DCN should choose the server or VM for aggregation that has a large computational and storage capacity. The downstream workers must also immediately communicate with the aggregator, which requires the DCN topology to reduce their hop distances to be most efficient.
- **Sequential workflow:**In this workflow, the application is processed in sequence. Future requests depend on the results of the previous work. For instance, a page load at Facebook [22] consists of at least 130 different data requests that often have sequential dependencies. The QoS of a page load is typically that it occurs in less than 300 ms [23]. Therefore, the DCN is responsible for obtaining a QoS response for hundreds of sequential requests from the application layer.
- **Parallel workflow:**In this workflow, multiple requests are linked to the same worker that generates the results and issues responses. Alternatively, an application may be distributed to multiple workers to be processed. For example, the “Shuffle period” [24] in MapReduce is a parallel workflow, where different Mappers send thousands of results to a Reducer. Therefore, a partition/aggregate workflow could be regarded as two parallel workflows and a sequential workflow. The DCN guarantees that all the flows produced by parallel workflows will be processed before their deadlines: otherwise, the parallel flows will generate incast problems in the network [25].

Since workflows from a CDS produce a large number of flows that have tight deadlines according to a cloud SLA [6], there are many challenges when deploying CDS in the currently available DCNs. First, existing DCNs do not adequately tackle the flow scheduling problem. In particular, most flows in a current DCN are short ($\leq$10 KB) [8,26]. On the other hand, long-lived, greedy flows are found to lengthen the bottleneck queue in the protocol stack until packets are dropped. That is because switches in the DCNs only provide per-packet scheduling where they are not aware of flow size and deadline. As a result, the short flows are usually starved when they are blocked by the long flows in a DCN switch.

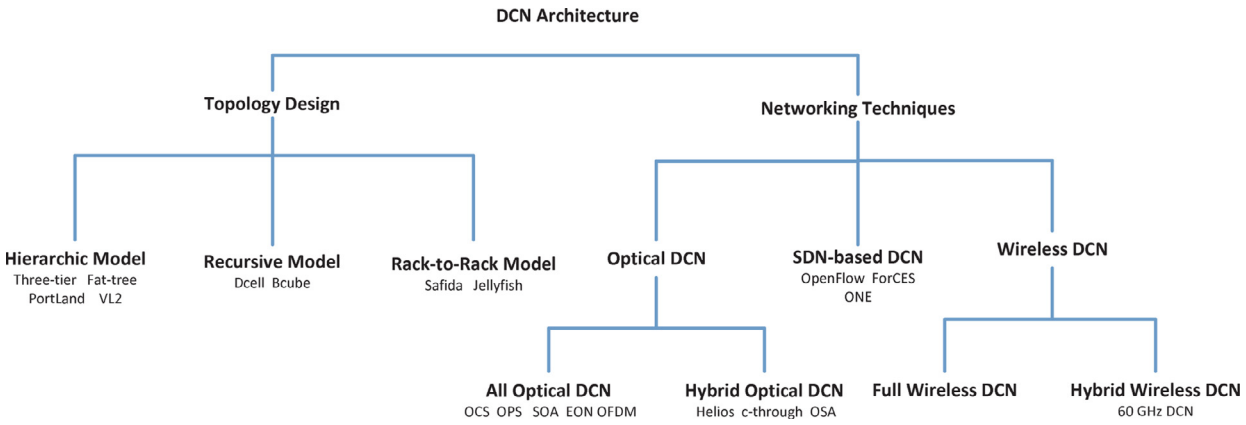


**Fig. 2.** The existing DCN architectures surveyed in this paper.

The second challenge comes from the fact that existing DCNs do not adequately balance their loads. A majority of traffic comes from the rack layer. Ref. [26] showed that the rack layer contributes 80% of data center traffic in a cloud data center and 40–90% traffic in a university or private enterprise data center. On the other hand, traffic hot is a great challenge at the rack layer. Ref. [27] showed that only a few Top-of-Rack (ToR) switches (5–10%) are traffic hotspots and most of their traffic goes to a few other ToRs.

Finally, in each workflow, CDS tasks assign the workers and aggregators in a random way. For example, many write behaviors in MapReduce are not assigned to a destination, so that the worker nodes at the next hop are unpredictable [28]. That causes challenges in the DCN routing such as routing loop problems. Additionally, since aggregator nodes require higher compute and storage capacities than worker nodes, assigning an aggregator in a poor computing environment could reduce the overall performance of CDS workflow.

## 3. Physical DCN architecture: topology design for cloud computing

DCN architecture lays out network elements and installs network techniques within a data center. DCN architecture is usually implemented from two directions: First, by designing a new topology that builds a cost-effective DCN to scale up the data center; second, by developing networking techniques within the existing topology so as to address the challenges due to the DCNs existing topology, as shown in Fig 2. This section provides an overview of DCN topologies and further describes their functionalities in supporting cloud service, while the next section will discuss the second direction: networking techniques to implement cloud DCN.

Recent research has shown that network topology requires improvement to remedy the congestion at trunk links, speed up end-to-end communication, and increase the network scale. We categorize the existing solutions for DCN topology as a hierarchical model, recursive model, or rack-to-rack model, and we discuss their impact on cloud computing performance.

### 3.1. Hierarchical model

Hierarchical model arranges network elements in multiple layers. Each layer characterizes traffic differently. This model is able to reduce congestion within the network because an upper layer switch prevents an overload of traffic that would otherwise all go through the same switch in a lower layer.

Three-tier DCN is the most commonly deployed hierarchical infrastructures, consisting of core, aggregation, and edge layers [7–9]. The switch at the edge layer (sometimes called the rack layer) connects to servers. The switch at the aggregation layer collects bits of communication from the edge layer, and conveys them to the core layer switch. The switch at the core layer connects to the Internet, and manages different traffic requests from the aggregation layer. Since the deployment process requires less hardware modification, Three-tier makes it easier to manage different virtual networks for tenant in the cloud.

Another hierarchical DCN example is Fat-tree [29]. In a DCN with Fat-tree topology, switches at the aggregation and edge layers are arranged in pods, where each pod is responsible for routing end-to-end communications. Switches at the core layer simply connect the egress switches of each pod. This reduces the traffic load overhead at core layer. Since Fat-tree DCN could use commodity switches at each level, it is highly cost effective. Similar to Three-tier DCN, building Fat-tree requires less both hardware and software modifications. PortLand in [30] addresses IP and MAC schemas for Fat-tree. Each PortLand server has an Actual MAC Address (AMAC) and a Pseudo MAC Address (PMAC). PortLand switches forward a packet based on its destination PMAC address, where PMACs record the topology information of a host.

Microsoft [21] has presented a DCN architecture called VL2 that is designed at layer 2 in order to provide an overlay infrastructure across different domains [31]. VL2 uses two IP address families: Location-specific IP Address (LA) for the server, and Application-specific IP Address (AA) for the application or VM. The LA is used to forward the packet in the physical network while the AA of each application remains unaltered regardless of shifts in a server's physical location. The sending server encapsulates the AA information in the

LA packet header, but it is trapped and de-encapsulated by the ToR switch associated with the receiver [13].

Hierarchical model is the mostly used architecture in DCN. The architecture shows many real world evaluation results and experiences for cloud service. Since today's most commodity servers, switches, routers support this architectures, there is not any large hardware and software modifications to build hierarchical DCN. Thus, most solutions of virual network and VM management apply to hierarchical DCN.

However, the design of most hierarchical DCNs does not take into account the scalability issues that they incur with their complex topologies when connecting massive numbers of servers. This feature does not apply to the server expansion of cloud computing. Besides, real world results [27,32,33] showed that hierarchical DCNs possess load-balancing problems. This will reduce the performance when the hierarchical DCN emerges as the infrastructure of big data system or CDS.

### 3.2. Recursive model

A recursive DCN consists of individual cells, each of which includes a single switch and a number of servers [8]. Servers bridge different cells, requiring operation at a much higher rate of computation than other servers. DCell [34] is a typical recursive DCN architecture that relies on cheap commodity switches. The goal of DCell is to exponentially scale machines with each iterated expansion of the network. Each server acts as a routing device, connecting other servers in different DCells. When traffic from cloud applications is generated, links connecting cells on the same level exhibit the same oversubscription ratio and traffic load [12]. Therefore, a DCell at a higher level generates a higher latency and congestion than one at a lower level. With traffic scale expansion, the inter-DCell link might become a bottleneck.

BCube [35] uses a switch instead of a server to connect different cells. BCube accelerates the one-to-all traffic, i.e., the traffic from one sender to all other receivers. The architecture applies a higher oversubscription ratio at the trunk link, e.g., 1:256 [8]. However, BCube exhibits poor incremental expansion in that servers must overwrite a shared address space and reconstruct the cabling topology in order to complete node updates.

Most recursive DCNs have high scalability to allow data center expansion and are cost-effective because they require cheaper switches than hierarchical DCNs (the switch is usually not used for bridging the routing in the network). These features create an opportunity to deploy large-scale cloud computing service with a recursive DCN. Most recursive DCNs have effective solutions for load balancing, some of which have symmetric features. Thus, a recursive DCN has an advantage for traffic-intensive cloud computing workloads such as for big data and CDS.

However, recursive DCNs are not commonly seen as the infrastructure for cloud computing, in part because of the lack of adequate field testing of their designs. In particular, most recursive DCNs employ a computational server to bridge the network routing. That requires the bridged server to not only offer all-to-all interfaces but also have higher capacity than servers in a hierarchical DCN. Therefore, the network interfaces of a bridge server will share resources such as CPU, memory and interface buffers with other VM instances. This sharing significantly reduces the server performance when they run VMs for cloud service.

### 3.3. Rack-to-rack model

Because the backbone link in the two models previously described is where bottlenecks typically occur, recent research into DCN architectures focuses on making a direct connection between different racks, as opposed to setting trunk layers.

In Ref. [33], a DCN topology called Jellyfish uses a random graph to build end-to-end communication. Instead of a fixed topology, Jellyfish networks simply guarantee that each switch provides r ports to other switches, while the remaining ports are connected to servers. Jellyfish better utilizes bandwidth than Fat-tree as it makes better use of effective path lengths than the latter. Similarly, a Jellyfish-based DCN lowers the mean path length that is computed by the sum of all path lengths.

Scafida is a rack-to-rack architecture that addresses large node degree within a DCN topology [36]. This architecture is a scale-free topology where the longest path has a fixed upper bound. This enables Scafida to artificially constrain the number of links from a server or switch, allowing for any size of data center to be created. However, Scafida has no fixed addressing or routing schemas. We believe that Scafidas forwarding table size could be incrementally expanded to suit the scale of its application.

Rack-to-rack DCNs provide a simple solution to achieve the interconnection of massive numbers of servers. Some rack-to-rack DCNs have significantly reduced the traffic complexity compared with hierarchical and recursive models. This feature enhances the network load balance when rack-to-rack DCNs are used for cloud computing infrastructure. Moreover, most rack-to-rack models support incremental expansion, where adding or removing one server does not cause a large modification of network topology. This offers a flexible infrastructure if the CSP desires to increase or decrease the cloud service in their data center.

Similarly to recursive DCN, most rack-to-rack DCNs lack real world deployment for cloud computing purposes. The address schema is a significant problem for the use of such DCNs in cloud computing. That is because rack-to-rack solutions provide a loose topology where the server is positioned in a random node location. This creates challenges for Internet routing that is needed for cloud computing. Finally, rack-to-rack DCNs for big data involve many redundant paths and loops, which reduce the network throughput and increase the overheads to the switch lookup tables.

### 3.4. Topology features for cloud service

A cloud system emerges as the pattern of distributed application workloads on the end hosts, in which the application layer information is agnostic to the lower layers in the DCN. Due to cloud computing existing only in this abstraction, it is hard to comprehensively evaluate the cloud performance of a DCN topology based on various cloud computing activities. However, we are still able to present some features for a DCNs topology when intended for cloud service, which could help optimize cloud computing performance:

**Table 1**
Topology features for cloud computing and the comparison of the existing DCN topologies.

| | Parallel traffic | Backwards | Load balance | Robustness | Auto. naming | Scalability (K: number of switch ports) | Oversubscription |
|---|---|---|---|---|---|---|---|
| Three-tier [7] | × | √ | × | × | Broadcast/Multicast | $K^3/4$ | 1/2.5 to 1/8 [37] |
| Fat-tree [29] | √ | √ | × | × | Broadcast/Multicast | $K^3/4$ | 1:1 [38] |
| PortLand [30] | √ | √ | × | × | LDP [30] | $K^3/4$ | 1:1 |
| VL2 [31] | √ | √ | √ | × | VL2 directory system [31] | $K^3/4$ | 1/240 [31] |
| $DCell_2$ [34] | √ | √ | × | √ | Heart-beat broadcast [34] | $(K+1)^4$-1 | 1/256 |
| $BCube_2$ [35] | √ | √ | × | √ | NHA hashing [35] | $K^3$ | 1/256 [8] |
| Jellyfish [33] | × | × | × | √ | – | N(K-r) (r: network degree, N: number of switches) | – |
| Scafida [36] | √ | √ | × | × | – | $K^3/4$ | – |

- **Parallel traffic:**Many cloud applications require all-to-all, one-to-all, or one-to-many communications. For example, the Map tasks periodically write slavers in Hadoop [20]. Thus, there is a need for a DCN topology to support these properties.
- **Bandwidth oversubscription:**This is an essential factor in the design of DCN topology, as modern DCNs are always oversubscribed between the server bandwidth and chunk link capacity. However, since the cloud SLA strictly defines the bandwidth oversubscription on each user, there is a need for the existing DCN to arrange an appropriate bandwidth oversubscription ratio in the topology.
- **Backwards capability:**A DCN should be designed for high backwards capability, in which reversing a source and destination does not affect the traffic performance. There are also various use cases of such a feature in cloud computing. For example, some VMs in cloud computing need the full bisection bandwidth in which the transmission rate remains unchanged after reversing the direction.
- **Scalability:**This feature has emerged as the most important factor in current DCN design, as DCNs must address the interconnection of thousands of servers into which cloud computing can expand its service. Also, there is also a need to allow incremental expansion capacity in the DCN design, where upgrading DCN equipment does not require substantial rearrangement of the topology.
- **Automatic naming:**A DCN topology should find an automatic process to self-learn an address space for a DCN [2]. In particular, using the proposed process, a DCN switch could automatically detect the DCN without offline manual configuration.
- **Robustness:**A DCN topology should provide robustness with respect to network traffic. For example, when some switches or links fail to transmit, the DCN should be able to choose a redundant path to ensure undisturbed communication between two cloud computing nodes.
- **Load balance:**A DCN topology should consider the load balance of DCN traffic. In particular, it should avoid bottlenecks involving the server connection. This is important because cloud computing applications generate large quantities of data and a bottleneck would increase the latency of data exchange.

With these suggested topology features, we can roughly evaluate cloud computing performance in several DCN architectures surveyed in this paper. Our conclusions are shown in Table 1, where the symbol "√" means high performance, while "×" means low performance and "–" means no consideration of this feature. Load balance is not well designed in most DCN topologies. That is because the features such as scalability and parallel traffic might increase the congested links if the network employs a poor routing or traffic engineering. On the other hand, scalability is the most considered feature in the design. In particular, the 2-level recursive model can support at least 3 times more servers at full capacity than hierarchic model. In term of the automatic naming, many DCNs presented their own broadcast/multicast mechanisms to find the node position or address. For example, VL2 eliminates the most common sources of broadcast using VL2 directory system [31]. Finally, DCNs could arrange different oversubscribed ratio to guarantee their cloud QoS. A large oversubscription ratio, such as 1:256 in DCell, is provided in order to augment concurrent communications. A small oversubscription ratio, such as 1:1 in Fat-tree, is usually provided to ensure size-independent QoS.

### 3.5. Future research direction

Although many DCN topologies have already been deployed for cloud service, the industry attempts to design a cloud-specific DCN for future cloud computing services. Cloud-specific DCNs are a subset of "cloud DCN" so far, and are optimized for cloud computing: in contrast, a cloud DCN is usually designed to be general purpose, e.g., for green computing or disaster recovery, but has also been used for cloud computing service. Cloud-specific DCN topologies offer elements that strictly follow the cloud computing models defined by NIST, and we believe that a cloud-specific DCN can offer the following benefits to cloud computing:

- Self-service provisioning.
- Flexibility in arranging and upgrading resources based on different SLAs.
- A centralized control plane with distributed computing capacity.
- Application-driven QoS control.

The first three of these characteristics meet the on-demand flexibility requirements of cloud computing. With self-service provisioning, a DCN can be considered an Operating System (OS) in which all the computations from cloud computing are regarded as the tasks running on the OS. The proposed networking framework enables the tenant to access functions for virtual network isolation, addressing, and

application acceleration. The DCN is responsible for scheduling different tenants requests to the appropriate CDSs. On the other hand, the DCN should have the flexibility to manage and upgrade its resources. For example, a DCN could quickly detect starved VMs and schedule residual resources, e.g., migrating these VMs to an idle server with low overhead. Both of these examples need a centralized control plane to coordinate different DCN devices. However, the centralized control plane has the drawback that it requires efficient scalability for thousands of DCN servers and VMs. The on-demand resource allocation requires high computational capacity in the control plane. Such a control plane is impossible to implement within the existing techniques such as OpenFlow [39]. The last characteristic of cloud-specific DCNs is that they should offer an application-driven QoS trace and control. For example, Cloud-specific DCN can be used for Information-Centric Networking (ICN) [40], a cloud computing instance in which content distribution is implemented directly into the network fabric. The key challenge is how to quickly deliver information from the application layer to an upper layer. A future solution will require an intelligent control plane and a specific agent on each end host.

## 4. DCN architecture: Networking techniques for cloud computing

By utilizing any of the aforementioned topologies, a DCN can be established for thousands of servers. However, without networking efforts from the industry, the design of a data center topology is limited by the following: (i) arranging the routing, (ii) reducing connective complexity, and (iii) providing enough flexibility for the traffic. In recent years, various networking techniques are presented to address the DCN topology challenges when arranging cloud computing service. This subsection highlights the respective techniques and discusses their use cases for cloud computing.

### 4.1. Optical DCN architecture

Optical DCN (O-DCN) is a DCN architecture that relies on optical switches and cabling. Because optical signals facilitate high-speed connections and low energy consumption, such networks achieve large bisection bandwidth and readily permit scale-up. O-DCN can be classified as all O-DCN or hybrid O-DCN, as shown in Fig. 2.

#### 4.1.1. All O-DCN architecture

In the all O-DCN, all the elements from control plane and data plane employ optical circuits and amplifiers. The goal of using all O-DCN is to offer high-speed bisection bandwidth to the DCN. In this respect, Optical Circuit Switching (OCS) DCN can provide large bisection bandwidth at the core layers [4] . The architecture is designed to pre-configure the static routing paths in the switch. OCS is effective even in a congested data center, as it uses per circuit processing instead of per-packet [44]. Another optical technique called Optical Packet Switching (OPS) [4] was proposed to offer on-demand bandwidth switching in the DCN. OPS is only efficient if the O-DCN offers all-optical buffers [45]. Likewise, Semiconductor Optical Amplifier (SOA) [46] based approach can arrange different broadband band behaviors, and alternatively configure switch for packet or circuit mode.

Elastic Optical Network (EON) [41] offers centralized on-demand flexibility in bandwidth switching. In an EON, the optical spectrum is divided into multiple fixed spectrum width portions. These portions could also be aggregated to slots and slot width, which are assigned to a path by EON control plane. An EON channel consists of numbers of slots and slot width. Therefore, EON control plane could easily switch the slot path using spectrum control on the fly. Recent research aimed to address Routing and Spectrum Assignment (RSA) [41] of EON, in which the same frequency slots in the optical signal must be assigned along the end-to-end path. Since most EON architectures compute and allocate a set of slots using centralized control plane, the EON-based O-DCN lacks the scalability for expanding large-scale cloud service.

Also, there is a need in all O-DCN that multiple low-speed traffic streams onto high-speed wavelengths in order to minimize switching overhead. The respective technique is well known as traffic grooming [47], which has already been used in Wavelength-division multiplexing (WDM) environment [48]. Orthogonal Frequency Division Multiplexing (OFDM) [42] is the most common modulation techniques, which is designed to improve bandwidth capacity for heterogeneous flows [9]. In summary, the main challenge of traffic grooming problem in an O-DCN is how to distinguish signals and properly adjust the filtering when switching a connection.

#### 4.1.2. Hybrid O-DCN architecture

Hybrid O-DCNs were proposed in order to provide extra bandwidth on demand with less impact on the existing network and to shift the connections in order to minimize hop routing. A hybrid architecture can avoid the congestion from large applications at rack level [32]. The multiplex feature of optical signal pathways helps to reduce the traffic complexity and provide on-demand connections [44]. In this respect, Helios in [43] uses mirrors mounted on a micro-electromechanical system to route the optical signals to minimize traffic complexity at core layer [43]. The links from pod layer switches to the core layer are laid with WDM technique. On the other hand, c-through in [32] is a platform that recruits end hosts to perform traffic monitoring. This architecture achieves the topology flexibility with a limited number of single-hop optical links. However, the two architectures do not minimize both electric routing and RSA overheads, in which the optical link traffic is affected by DCN topology. Therefore, OSA in [27] builds one-hop communications and congestion control. An OSA controller consists of electrical part and optical part. The electrical part stores the routing information normally. The optical part is in charge of one-hop communication. In case of a congestion, the OSA controller could disable the congested link, establish one-hop connection from the source to the destination, and amplify its link capacity for the instantaneous demand.

Table 2 shows the existing O-DCN techniques and their traffic properties in cloud computing service. In this respect, EON, c-through, and OSA use centralized control plane to collect optical information and assign routing path to DCN traffic. In particular, c-through and OSA control planes are hybrid with both electric and optical part controller. However, the

**Table 2**
Comparison of the optical DCN architectures.

| | Control plane | Electrical part | Optical part | Drawbacks |
|---|---|---|---|---|
| OCS [4] | Distributed optical control | Topology-free | Optical circuit for static routing | Lack of flexibility |
| OPS [4] | Distributed optical control | Topology-free | Optical switch for dynamic routing | High-speed buffer requirement |
| EON [41] | Centralized optical control | Topology-free | Elastic optical controller | Lack of scalability |
| OFDM [42] | Distributed optical control | Topology-free | Traffic grooming | Long switching time, signal attenuation |
| Helios [43] | Distributed optical control | Core layer | Pod layer for large bandwidth | Large switch modification, poor optical fiber utilization |
| c-through [32] | Centralized hybrid control | Fat-tree | Optical controller connects to all ToRs | Lack of scalability, poor signal synchronization |
| OSA [27] | Centralized hybird control | Topology-free | Optical circuit for one-hop connection from the source to the destination | High component cost, long sleep time |

centralized control plane has poor scalability if the O-DCN is used for large-scale cloud service. On the other hand, the distributed control plane techniques such as OCS, OPS make the local decision in assigning RSA path. But most of them require either large modification or high-speed buffers to the DCN switch. Therefore, building such O-DCNs incurs high cost.

#### 4.1.3. Opportunities and challenges for cloud service

Optical technique has the pros to reach many cloud computing requirements. In particular, it offers high-speed traffic with low energy consumption and low overhead than electric technique on the link. Research efforts in this field have proposed various solutions to ensure the large-bandwidth traffic using optical technique as it emerges as a backbone technique used for Internet fiber. Besides, some optical techniques such as EON, OFDM provide the multiple channels and the flexible switching solutions. This makes them easy to manage bandwidth resource on demand in the cloud service. Finally, RSA algorithm from the optical control plane can significantly minimize routing complexity than electric routing approach, in which all path assignments are processed by switching optical signal. This offers the chance that some routing decision could be decoupled from the electrical switch to the optical control plane.

However, using optical technique in the DCN for cloud computing should address the following challenges. Firstly, due to the lack of high-speed optical fabrics and all-optical buffers, the existing DCNs have limited capacity to deploy optical switches. Even all the cabling of DCN is replaced with optical fiber, it incurs high cost in order to expand cloud computing service to thousands of users. Secondly, since many optical techniques offer the large-bandwidth to DCN end-hosts, there is a need for these hosts to fully utilize the bandwidth such that an O-DCN can achieve high throughput. Also, we believe that some multiplex techniques such as EON or traffic grooming can adjust the available bandwidth to arrange the RSA path in the future. For example, the EON control plane need information of frequency slots availability such that it can arrange more streams on these slots [44]. Finally, in some hybrid O-DCN, the time interval of optical switching does not reach the full bisection requirement, this make it low efficiency to arrange routing and traffic to the urgency applications in the cloud.

### 4.2. Wireless DCN architecture

The challenges faced by both electrical and optical networks are primarily the manual cost and overhead of the cabling for large-scale networks and the required wiring changes to every ToR in a network. Increasing scale, routing, deployment, and repair of tens of thousands of wires incurs a substantial cost [4]. Moreover, a topology distributes a fixed number of wires, where networks are typically designed for a worst-case communication scenario. Consequently, Wireless DCN (W-DCN) has been proposed to address the aforementioned challenges. 60 GHz [49] W-DCN is currently the most commonly used technique, in which 60 GHz radios are placed on the top of each rack to connect pairs of ToR switches. A 60 GHz link can use beamforming, a physical layer technique that concentrates transmission energy in a specific direction to enhance the link rate and further suppress interference. In general, the existing W-DCN could be classified as hybrid W-DCN or full W-DCN, as shown in Fig. 2.

In a hybrid W-DCN, the whole topology usually follows a hierarchical model, in which the core and aggregation levels employ wired commodity switches, while the ToR level uses wireless technique. Each ToR has a wireless launcher and acceptor, which are responsible for end host routing with beamforming. Also, there are other efforts to enhance the performance of hybrid W-DCN in the ToR. In this respect, 3D beamforming in [50] establishes non-blocked 60 GHz connections in which 60 GHz signals bounce off ceilings in the W-DCN room.

A full W-DCN usually follows the topology of a rack-to-rack model, which includes the design of specific ToR switches for acquiring the wireless signals from other ToRs. Each rack has the responsibility to bridge the routing from other wireless ToRs [4]. Since the rack-to-rack model is capable of computing multiple redundant paths for each ToR, the full W-DCN shows high fault tolerance. Moreover, since each wireless ToR flexibly considers keeping or deleting the routing path to others on demand, the full W-DCN overcomes the aforementioned challenges of redundancy in the rack-to-rack model, such as the possibilities of forwarding loops and low link utilization.

Although a W-DCN can reduce traffic complexity and solve the bottleneck, there are many challenges to implementing a wireless DCN for cloud computing. First, most W-DCNs have low transmission rate (bandwidth) compared to a wired DCN, since 60 GHz only provides a maximum rate of 2.5 Gbps [49]. This rate does not meet the high-rate delivery requirements of cloud computing. Second, as the wireless signal is available only over a short distance (e.g., 10 m) due to its strong attenuation, it creates challenges for ToR placement. Finally, the wireless signal suffers interference from

many factors, including physical barriers, weather, and temperature, which affect the stability of transmission in a cloud service.

### 4.3. Software defined networking in DCN

Since a CSP can offer VMs and virtual networks with topology and QoS customized to the users' demands, there is a need for a cloud DCN to provide intelligence in the control plane and traffic flexibility in the data plane. SDN technique offers a powerful way to achieve these goals by decoupling the control plane from the device and centralizing them in order to manually inject and shift control logic [51,52]. This enables the administrator to configure the network hardware directly from the controller, thereby enabling many communication and traffic activities to be implemented with the help of SDN techniques. An SDN-based DCN is a DCN architecture that employs SDN switches and controllers. Since various surveys [51–54] have been conducted to overview SDN techniques and their use cases, this paper is only concerned with the functionality and challenges of using the SDN technique for cloud computing.

#### 4.3.1. SDN standardization efforts

SDN standardizations from different work groups usually follow the controller-service and controller-switch modes, where the northbound interface [53] provides the data exchange between controller and applications, while southbound interface [54] provides the communication between data plane and controller. Usually, these standardizations are not compatible with each other. This requires that the standardizations of SDN switches and controller in the DCN do not conflict [52,90]. The usual seen SDN standardizations which have been used in DCN are explained below.

The Open Networking Foundation (ONF) [55] developed OpenFlow [39], the most useful SDN standard. The OpenFlow standard includes, among others, the OpenFlow controller, OpenFlow-enabled switch, OpenFlow channel, and OpenFlow protocol, all of which have various implementations that are widely used in today's enterprise DCNs [52]. In each data plane, the OpenFlow switch handles a flow table that stores a list of rules for packets. When a packet arrives, the controller first queries the flow table for forwarding. If the target address does not hit the flow table, the control plane decides the next forwarding strategy and overwrites the flow table. Otherwise, packets will be sent normally according to the flow table. To better suit cloud computing applications, the OpenFlow protocol has defined a multi-tenant virtual network service.

ForCES [56] is a distributed SDN technique that combines new forwarding hardware with third-party control within a single network device [52]. The control and data planes are decoupled but kept within the same box or rack. The box is called the Network Element (NE). In particular, the NE consists of the Forwarding Element (FE), and the Control Element (CE). Similar to the OpenFlow switch and controller, the FE provides per-packet forwarding while the CE employs the ForCES protocol to instruct the FE's forwarding. The FE relies on Logical Function Blocks (LFBs), which play a role like a flow table in an OpenFlow switch. Outside of an NE, there are two managers, the FE manager and CE manager, which pre-configure all the FEs and NEs, respectively. The ForCES standard is quite different to OpenFlow because it decouples the controller within the NE, thus creating box-based management for the whole network.

ONE is an SDN project proposed by Cisco [57], which aims to centrally manage hybrid resources such as a 60 GHz wireless network and mobile network. Cisco ONE enables flexible, application-driven customization of network infrastructure to help realize business objectives such as increased service velocity, resource optimization, and faster monetization of new services [57]. ONE delivers data through a rich set of platform APIs, agents, and overlay network techniques [57]. ONE extends the optimization targets from OpenFlow to the entire solution stack from multiple layers.

The industry has recently undergone a significant transformation of its network infrastructure to extend the SDN to support multiple standards. The OpenDaylight project provides a plug-in mode for various southbound and northbound protocols [58]. The proposed SDN architectures include Network Service Function (NSF) and Service Abstraction Layer (SAL), which are tolerant of being mixed with other protocols. For instance, NSF can translate communications from different cloud applications for the controller, while SAL is responsible for configuring protocols from different SDN switches in a southbound chip. To this end, the DCN need not coordinate the conflict between controller and switches with different SDN standards.

#### 4.3.2. SDN use cases for cloud computing

The primary use case of SDN in this context is to build a flexible routing scheme [53]. An example is the communication between the OpenFlow controller and the switch, where the flow table is updated through the control message in a short period. This offers an efficient way to reduce switching consumption among different routing solutions. Thus, a SDN-based architecture could be deployed in a DCN for routing the random write behaviors (e.g., Shuffle period in MapReduce) in the cloud.

Since SDN offers the promising solution of a message tunnel from the controller, a switch need not self-learn all of the network topologies. This relaxed requirement enables the DCN to improve the performance of VLAN-based communications at any scale of topological complexity. SDN-based VLAN infrastructures, such as Contrail [59] and NSX [60], might emerge as a feasible infrastructure to provide multi-tenant isolated virtual networks.

SDN provides efficient global control and a programmable interface [54]. Therefore, various traffic management issues could be implemented through the SDN-based architecture. In this respect, the work in Ref. [61] uses OpenFlow to reroute flows when the path node is updated. The flow table entries provide a customized configuration while the controller computes the path information based on WCMP using a fat tree structure.

#### 4.3.3. Open research issues of SDN

Despite the various benefits of using SDN for cloud computing, there remain several challenges that can restrict the real-world utility of SDN in a DCN. They include the following aspects:

- Using SDN to build a cloud DCN is costly. For example, both OpenFlow and ONE switches are ten times more expensive than a commodity switch that could be used for building a BCube or fat tree. This is because the hardware on the southbound and northbound interfaces is more expensive than commodity switch chips. Since today's enterprise DCNs are required to be built in a cost-effective way, it is impossible to replace thousands of commodity switches in the DCN with SDN switches. We believe that future SDN architectures for cloud computing will be a hybrid in which only the switches bridging the most DCN traffic will be controlled with the SDN control plane. This requires the northbound API in the control plane to efficiently collect the information from non-SDN switches in order to send control messages to the SDN-based switches.
- Besides the cost-effect conflict, DCN scalability also affects the deployment of both centralized and distributed SDN-based architectures. In a centralized SDN-based architecture, such as ONE, it is hard for a centralized control plane to inform and control thousands of data plane switches in the DCN. In a distributed SDN architecture, such as ForCES, the system lacks coordination among different controllers in the large-scale DCN because the northbound control plane makes the local decision in a small area. We believe that future SDN networks should be designed to (i) follow a heuristic placement algorithm that determines how many and the positions of SDN controllers, (ii) use a special tunnel for the communication of different controllers, and (iii) use a centralized control plane to coordinate different controllers.
- Although SDN offers an implementation that addresses many DCN challenges in cloud computing service, the existing SDN techniques lack the reliability to provide optimized QoS on demand in the cloud. For example, the on-chip OpenFlow forwarding rules in commodity switches are limited to a small number, typically 1–4K. This will cause frequent updates from the control plane when burst traffic congests the switch. On the other hand, the optional modification of the flow table will undermine the automatic actions of the SDN switch, for instance, topology self-learning and packet filtering [52]. This might affect the normal function of these switches to bridge end-to-end communications. Thus, future research should address these problems.

## 5. Virtualized infrastructure from DCN for cloud computing

VI is a network infrastructure where some or all of the elements (e.g., servers, links, switches, topology) are virtualized by the DCN, as shown in Fig. 1. The cloud computing platform consists of single or multiple VIs, which rely on virtualization techniques to partition available resources and share them among users. This section describes the architecture of VIs using DCNs and their suitability for cloud-based service. Finally, we present open research issues associated with VI.

### 5.1. Virtualization techniques used for VI

DCN technique promises an interactive solution to implement VI and cloud computing within existing virtualization techniques. Currently, the following types of virtualization techniques are used with DCNs to build a VI for cloud computing:

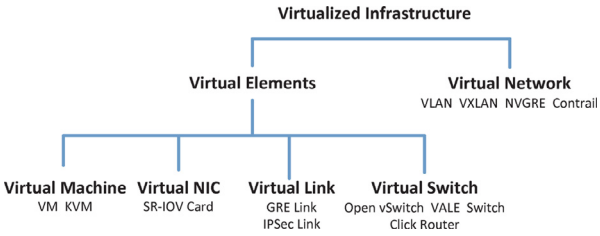


**Fig. 3.** The virtualized infrastructure for cloud computing.

- **Server Virtualization (SV)**: A piece of physical hardware is virtualized using software or firmware called a hypervisor that divides the equipment into multiple isolated and independent virtual instances. SV solves the problem of VM allocation in cloud computing. Since a VM instance is created and placed in its host using the intelligent solutions of a cloud system (e.g., Swift), a DCN should ease the communication between different hypervisors in order to quickly complete the allocation [5].
- **Network I/O Virtualization (NIOV):**NIOV provides an abstraction that shares network I/O devices between VMs. The most commonly used NIOV in the cloud DCN is the NIC virtualization, in which physical NIC is shared among multiple VMs through the hypervisor [62]. To serve cloud computing needs, a DCN uses NIOV to provide large bandwidth capacity and low CPU overhead access to each VM in the virtualized I/O device.
- **Network Virtualization (NV):**NV enables the creation of logically isolated virtual networks over the physical DCN. Therefore, NV solves the cloud computings requirements for VI topology, addressing, and routing architecture [54].
- **Resource Virtualization (RV):**RV solves the VIs requirements for resource allocation and management of VMs. A DCN should address the problem of VM placement such that each VM does not oversubscribe the SLA resource [63].

### 5.2. VI architecture

A VI architecture is made up of VMs, virtual NICs, switches (either the physical or the virtual) and links (either the physical or the virtual), as shown in Fig. 3. Most of them are instantiated over real machines of DCN. The virtual network is the network used for the communication in VI. This subsection will discuss the different elements and the used techniques in the VI architecture.

#### 5.2.1. Virtual machine

A VM is started and managed by a hypervisor, which provides a binary translation service on shared hardware (typically a x86-based machine) [62]. An existing hypervisor, such as Xen or VMWare, can rapidly provide a VM image with large CPU and disk capacity. Since traditional SV technique, such as full virtualization and para-virtualization, incurs high latency during binary translation, some lightweight VMs, such as Kernel VMs (KVMs) [64], are now being deployed in cloud

DCNs. The benefits of lightweight VMs are that they can reduce the translation overhead from the hypervisor, so that a host can start more VM instances than traditional VMs with less CPU consumption.

Despite the maturing of this technique, creating VMs for on-demand allocation and use is still a challenge to existing SV technique. In particular, there is still a need to efficiently utilize the CPU and image space in VMs with large CPU and image capacity. For example, during the "Shuffle period" of MapReduce, the hosting VM runs CPU-intensive tasks and thus requires large cloud CPU capacity [91]. However, when it shifts its role to become a slaver VM, the CPU capacity could be wasted because there is not enough CPU-intensive computing assigned to it.

#### 5.2.2. Virtual NIC

Virtual NIC (vNIC) is a software emulation of a physical NIC, initially designed as an embedded module assigned to the VM by the hypervisor; however, the hypervisor must frequently intercept system calls from a large number of vNICs, so I/O control must be decoupled from the hypervisor. Techniques such SR-IOV [62] have been proposed recently to address this. SR-IOV is a technique that centrally manages multiple I/Os in a physical NIC. Each hypervisor corresponds to an SR-IOV card that virtualizes all the vNICs. A Virtual Function (VF) is assigned to a virtual NIC, including resource allocation and migration. When the remote VM accesses the domain, the SR-IOV card interprets the access information and maps the request to the target virtual NIC. Challenges of SR-IOV technique include only one server virtualizing one SR-IOV card and poor compatibility among different versions of cards.

#### 5.2.3. Virtual switch and virtual link

Virtual Switch (vSwitch) is a software tool, installed on a server or rack, which acts like a physical switch. vSwitch performs functions such as switching, multiplexing, and bridging to the physical NIC [5]. The vSwitch port for each active VM listens to the transmit buffer from the virtual NIC, while controlling packet loss based on a threshold placed on the queue. vSwitch forwards packets from input ports to output ports using a forwarding table. If the destination is determined to be local, respective packets are moved to their corresponding ports; otherwise, they are queued in the port corresponding to their physical interface. The most common vSwitch products are VALE [65], Open vSwitch [66], and Click Router [5]. vSwitch has the benefit of allowing customization of routing control (such as by enabling an OpenFlow module), augmentation of the layer 2 fabric, and flexible port management.

The virtual link is a special individual tunnel that provides traffic isolation to each adjacent VM or vSwitch. This tunnel may appear, for some protocols, to directly connect to the network in a similar way to a physical link. The virtual link is characterized by techniques such as GRE [67], and IPSec [68]. In addition to vSwitch, the virtual link relies on the CPU's processing speed in the server.

vSwitch and virtual link provide software solutions that reduce the burden from the physical switch and link. However, their performance relies on the computational abilities of the host server, where the data exchange from the virtual ports might occupy excessive memory. For example, the queue on each virtual port will allocate memory during the initialized period. Since the DCN connects thousands of servers and VMs, the workloads on each switch or link might be heavy. Under such circumstances, it may be impossible for the vSwitch and virtual link to finish the data transmission and exchange.

#### 5.2.4. Virtual network

The evolution of virtual networks can be characterized by two distinct time periods in which it has existed. The first is the pre-cloud period, during which time cloud computing had not yet been proposed. Techniques such as Virtual Local Area Networks (VLAN) [69] and Virtual Private Networks (VPN) [69] existed then, supported by almost all current layer 3 switches. However, virtual networks at this time could not fulfill large-scale demands for groups because they were still coupled with physical networks. The second period is post-cloud, when cloud computing became prevalent. To accommodate large numbers of virtual networks, techniques such as VXLAN [70], NVGRE [71] were proposed. In addition, techniques during this period, such as Contrail [59], have prioritized improved management and independence from their physical network.

VLAN provides logical isolation between broadcast domains at layer 2 by creating virtual networks. Multiple VLANs are bounded by differing segments [69]. The primary function of VLAN is to ensure that all the virtual networks are able to share the same physical infrastructure while being properly segmented. VLAN!'®s main advantage is its ability to isolate tenants' logical networks. However, because of the limited size of its segment (12 bits,) it is only able to create 4096 virtual networks.

VXLAN employs tunneling via MAC-in-UDP [63] to restrict packets in the destinations net segment. By using UDP multicast instead of broadcast, VXLAN significantly improves the migration performance among data centers while balancing the tradeoff with local broadcast. Moreover, VXLAN enables the cloud to create more virtual networks due to its larger segment address size (24 bits). The drawbacks of VXLAN include the requirement of layer 3 device support and the latency associated with long distances in a virtual network.

NVGRE applies GRE as a method to tunnel layer 2 packets across an IP network, and uses 24 bits of the GRE key as a logical network discriminator. The broadcast of the virtual network is achieved through physical multicast.

Contrail supports virtual networking using SDN-based virtualization, in which a controller can reset different encapsulation approaches in order to manage different layers virtual networks, e.g., layers 2 and 3 VPNs. The main functions of Contrail include abstracting network control into different layers, providing APIs to different versions of devices, and isolating tenants in the cloud.

Table 3 compares the existing virtual network techniques surveyed in this paper. In terms of their encapsulation, VLAN is the first that uses encapsulation technique to isolate different virtual networks. However, VXLAN and NVGRE employ faster tunnels for encapsulation than VLAN, because both UDP [72] and GRE perform better than IP in VM migration. An OpenFlow controller enables Contrail and NSX to

**Table 3**
Comparison of different virtual networks in the DCN.

| Virtual networks | Detailed description | Encapsulation | Packer header | Maximum VNs | Multiple tenants |
|---|---|---|---|---|---|
| VLAN [69] | Bridging different VMs and bounding the virtual network by differing segment | MAC-in-IP | 12 bits VLAN segment | $2^{12}$ | × |
| VXLAN [70] | Using UDP multicast instead of broadcast at layer 2 | MAC-in-UDP | Extending segment size from VLAN | $2^{24}$ | √ |
| NVGRE [71] | Using 24 bits of the GRE key as a logical network discriminator | MAC-in-GRE | Extending segment size from VLAN | $2^{24}$ | √ |
| Contrail [59] | OpenFlow controller to manage the virtual network | MAC-in-UDP, MAC-in-IP | OpenFlow segment bits | $2^{12}$ | √ |

manage common issues by monitoring VM flows and configuring different routing protocols. Column 5 shows the maximum number of virtual networks that a scheme scales. VLAN,and Contrail have a 12 bit segment which virtualizes up to $2^{12}$ virtual networks. VXLAN achieves high scalability by extending the segment size to 24 bits. All the techniques in the table except VLAN support multi-tenants, due to their cloud computing-optimized designs.

### 5.3. Resource allocation in VI

The goal of VI resource allocation is to build a VDC that allocates and manages targeted resources using RV technique. VDC is a collection of VMs that logically compose other isolated data centers. This type of network is established along with VM placement algorithms when the total VMs fulfill a resource requirement [73,74]. A VDC provides a minimal bandwidth to each VM that defines the worst possible performance of a tenant.

#### 5.3.1. VDC and virtual network

VDC and virtual network are complementary concepts that do not depend on each other. Unlike a virtual network, a VDC-based network does not use 2 or 3 layer techniques to build privileged communications among different VMs. Instead, a VM and a neighbor in the same VDC can be physically remote provided the two VMs are placed in a way that allows them to attain sufficient resources. Similarly, in the context of a virtual network, the address of each VM must be either overwritten across the physical infrastructure or encapsulated in the packet header, while a VDC-based network does not provide any address restriction to the member VMs [12,73]. Moreover, the communication elements in a virtual network include not only the VM but also the physical server and switch, while a VDC only guarantees resources for VMs.

#### 5.3.2. VDC architectures

Various VDC architectures have been proposed for VI resource allocation. The hose model in [12,73–75] is the most widely deployed VDC infrastructure. This model adopts the centralized approach, wherein each VM is directly connected to a non-blocked switch. A VM can only be blocked in the access link while, logically, each VM pair has a two-hop distance. The shortcomings of the hose model include substantial pressure from incremental expansion and non-oversubscription for bandwidth sharing in each server.

Oktopus in [73] shares many characteristics of the hose model but provides two kinds of clusters that address different problems. The virtual cluster is designed as a hose model for a group, supplying maximal oversubscribed bandwidth to each VM. Oktopus does not provide any bandwidth oversubscription to the group, so that the actual bandwidth requirement never reaches the aggregate value, effectively designing an oversubscribed cluster. Each group will have an oversubscribed factor that constrains the upper-bound bandwidth of each VM. Oktopus is the most commonly used network for intra-tenant communications. About 20% of inter-tenant traffic exists in cloud computing [17], however, which also generates unfair bandwidth consumption between the two kinds of clusters.

The hose-compliant model in [17] groups tenants so as to accommodate inter-tenant demands. Each tenant attains a minimal bandwidth for worst-case communication scenarios in the cloud. Each group also serves a tenant. The allocation policy follows a max-min fairness model [75] that allows a VM to expand the bandwidth. However, all the bandwidth values in the hose-compliant model eventually reach an equilibrium. Therefore, a VM cannot optionally increase its bandwidth while congesting others. The drawbacks of the hose-compliant model include the difficulty of its implementation, and the fact that allocating minimal bandwidth based on payment allows for network abuse of the expensive tenant.

#### 5.3.3. Virtual machine placement

VM placement is an RV technique that determines how a VDC is mapped to the physical infrastructure. Among other requirements, the mapping of a VDC must: ensure a minimal bandwidth to each VM for worst-case communication, ensure an appropriate oversubscription ratio, abstract the upper physical layers into a non-blocked switch, and limit interference with other VDCs. The oversubscription ratio is desirable because communications between VM pairs never consume their full bandwidths in a best-case scenario [73]. Similarly, a VDC does not consume excessive bandwidth while interfering with the performance of other VDCs in the same physical infrastructure [92]. In this respect, SecondNet in [12] places different VMs to guarantee bandwidth. A central controller determines the desired bandwidth and path for each VM by building a bipartite graph between the VDC and the physical infrastructure. Similarly, Oktopus aims to meet tenant's requests and enforces a virtual cluster with a greedy algorithm [73]. However, the two approaches use a fixed bandwidth reservation that can potentially waste

bandwidth when demand shifts. Additionally, both approaches assume sufficient physical resources that threaten to reduce link utilization, and they both assign a centralized allocator that increases calculation overhead with scale. In Ref. [74,76], the over-reserved bandwidth from fixed allocation allows more jobs to be admitted into the data center. These approaches ensure the benefits of good network utilization but ignore the enforcement of bandwidth limits that potentially congest other VMs in the same link. ElasticSwitch in [77] provides a max-min fairness guarantee for each links bandwidth, in which the controller rate-limits all the flows through each VM so that the minimal bandwidth of any other VM is not affected by its allocation.

Although placing VMs within physical infrastructure is an NP-heavy problem [12,73], the algorithms used by the aforementioned approaches are mostly heuristic and balance many tradeoffs. However, cloud applications are divergent in nature, and some are mission-critical for users. Thus, future placement algorithms must address relevant challenges such as the deployment overhead from the allocator and rate-limiter, necessary scheduling for over-reserved bandwidth, and reliance on a fixed physical topology.

### 5.4. Future research direction

There is a wide range of open, critical challenges to the design of a DCN that can support a reliable and efficient VI for cloud computing. This subsection will present some of these open research issues.

#### 5.4.1. Network function virtualization and VI

Network Function Virtualization (NFV) [53,54] decouples network functions, e.g., the firewall, load balancer, and Domain Name Service (DNS) from the physical server. NFV entrusts the functions running on Commercial Off-the-shelf (COTS) [53] equipment, e.g., a switch, to a dedicated machine for a Intrusion Detection System (IDS) [78]. It also builds Virtualized Network Functions (VNFs) that handle specific functions running in VMs on top of the COTS equipment. NFV offers central function management across the VI, in which each VNF is isolated. Also, NFV saves time on manual training and provisioning of cloud service.

However, the DCN meets several challenges when installing NFV for cloud computing. First, NFV introduces additional COTS equipment into the DCN architecture. The DCN should have an architecture tailored for this equipment to provide a central virtualization tunnel. For example, the equipment is required to take fewer hops to distribute its function to the VNF on the respective VM. Second, there is a need for the DCN to provide large bandwidth capacity to the COTS equipment such that it can scale more VNFs from more VMs. Third, the VNF could also be asked to migrate to another VM, but the current migration approach incurs high latency [9]. For example, if a firewall instance is required to move from one VM to another, it must stop all its working processes and keep snapshots of their state that must be duplicated and delivered to the destination VM. Thus, how to reduce this migration overhead requires further exploration.

#### 5.4.2. Container virtualization and VI

Container Virtualization (CV) [64] offers the concept of container, a more efficient virtualized element than VM that can run a variety of applications in the cloud. Traditionally, each VM corresponds to a single kernel virtualized by a hypervisor on top of a host kernel. If the cloud application running on the VM accesses the host kernel, the system call from the hypervisor would incur high CPU overhead because the call must be translated by the host kernel. In contrast, a container can share the host kernel directly without any translation, resulting in a faster application process. Presently, Docker is the most popular CV product.

Since CV is a new technique that removes the hosts need for a hypervisor, there are many open questions regarding the deployment of Docker for cloud computing over a DCN. First, current CV technique does not adequately address the performance of NIOV and NV. For example, the NIC cannot be shared efficiently by the container, VM, and KVM. This makes it challenging for a DCN to provide adequate bandwidth and I/O control to the container. Second, since the container is meant to replace the VM as the service unit in cloud computing, a DCN should efficiently allocate hardware resources to a different container according to the SLA. Thus, future research should consider the container placement algorithm as well as VM placement for different users. Third, in a hybrid environment where a container and VM coexist to consume a DCNs resources, how should the DCN manage their interaction and avoid resource allocation conflicts between them?

#### 5.4.3. Network as a Service and VI

Network as a Service (NaaS) [53] has emerged as an Internet-based service model for cloud computing that creates a number of VIs for on-demand resource provisioning to the tenants. Compared with VDC-based resource provisioning, NaaS incurs a higher level of demand from multiple tenants, but both concepts can be merged to flexibly utilize DCN resources during cloud service [93]. The primary goal of VDC is to address the resource allocation problem of one VI in which the allocator places different VMs to the physical infrastructure on demand, while NaaS allows multiple VDCs to share and manage DCN resources among different VIs.

DCNs for NaaS should not only address the intelligent VM placement problem intra-VDC, but also offer an intelligent VDC placement policy to coordinate the isolation and interaction of different VDCs. With existing DCN designs, it is hard to achieve on-demand provisioning in both intra-VDC and inter-VDC because both problems are NP-hard. However, there are also many opportunities if the VDC allocator heuristically considers multiple VDC demands when placing VMs to the DCN, rather than considering both VM and VDC placement problems individually. For example, the different SLA matrixes from different VDCs could be merged as a single matrix, so that the whole NaaS infrastructure could emerge as one single VDC. This would permit many VDC solutions to address the challenges of NaaS.

## 6. DCN routing for cloud service

Many cloud computing applications require fast DCN routing that minimizes the number of hops over the underlying topology. The main challenge inherent in DCN routing is arranging appropriate paths for two communication hosts within thousands of connections. The DCNs routing for cloud service can be designed from two perspectives: physical DCN

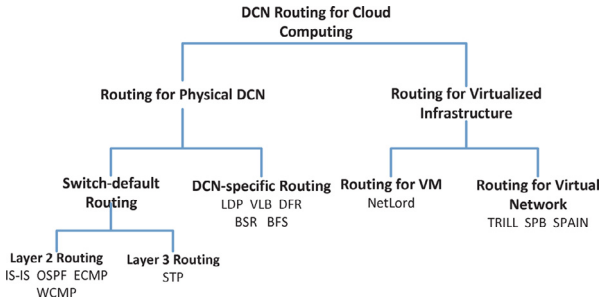


**Fig. 4.** The taxonomy of DCN routing schemes.

routing and VI routing, as shown in Fig. 4. Physical DCN routing concerns the routing from two physical hosts in the DCN, in which all the virtual layer activities (e.g., VM, virtual network) are regarded as the workloads. In contrast, VI routing provides the path computation for VMs and a virtual network. This section discusses both general routing use cases and the DCN-specific routing for cloud service.

### 6.1. Routing scheme for physical DCN

The routing for physical DCNs can be performed either using a switch's default routing protocols or DCN-specific routing schema. We now explain their features and cloud computing functionalities.

#### 6.1.1. Switch-default routing protocols

Since most DCN architectures employ commodity switches as the equipment to bridge server-to-server communication, the default configuration of these switches can be used to install general Internet and Ethernet protocols. In particular, to provide cloud service on the Internet, the whole data center is regarded as an Autonomous System (AS) [11]. Thus, many Interior Gateway Protocol (IGP) [79] protocols, such as IS-IS [80], OSPF [81] could be installed in the DCN switches for routing. In recent years, since many Multi-path Label Switching (MPLS) [82] switches are employed, the DCN could also simply apply multi-path protocols in its routing activities. The flow level multi-path protocol distributes a flow from one input port to multiple output ports, based on the residue bandwidth from the next hop in the DCN. The packet level multi-path protocol provides multiple output ports that have available buffer size for a packet. Two commonly seen multi-path protocols of DCN are Equal Cost Multi-path (ECMP) [83] and Weight Cost Multi-path (WCMP) [61]. On the other hand, a switch at layer 2 could employ Spanning Tree Protocol (STP) [84] for multicast and broadcast such that it can find its neighbors and determine the topology.

However, using switch-default protocols to bridge the cloud computing communications creates two key challenges. First, most of these protocols are not practical for DCNs because they are designed for small-scale local networks. Since the scale of DCNs used for cloud service is usually large, it is challenging to connect thousands of servers, compute paths, and perform switch learning by the switches and routers in the DCN. In a large DCN, this equipment must handle large forwarding table entries and may incur address resolution overhead. Second, switch-default protocols do not consider the topology of a DCN. When it performs switch self-learning or is manually configured, the next hop link or switch determined by these protocols may be a point of congestion or may lead to loops in the routing path. These problems will cause packet loss and increase traffic latency.

Although most of the general protocols are not used for routing cloud communication, some of them could still be used to address other QoS issues of DCN traffic. In this respect, Table 4 shows the use cases of general routing protocols in some example DCN architectures. Most multi-path protocols could be modified by a load balancer to reroute the flows from cloud applications. That is because these protocols can be simply installed and uninstalled in the switch, such that the developers need not make many modifications to the control plane logic of the network. For example, ECMP could be installed to provide a redundant path in [10,85]. The cited work addresses the drawbacks of ECMP and how it achieves high performance in some specific cloud scenarios.

#### 6.1.2. DCN-specific routing scheme

Most DCNs address elements and arrange routing based on their own topology [8]. The routing characteristics of typical DCN architectures are shown in Table 4. The details are as follows.

The recursive DCNs, such as DCell and BCube, use a cell to bridge between the routing and address servers. The prefix length of a server-centric address is changed with the recursive number. BCube builds its routing path by shifting one bit at one hop [35], but the calculation takes multiple hops beyond the shortest path. Conversely, the hierarchical DCN has more agility to design its routing protocol. For example, VL2 uses a bouncing switch that calculates only one path for any communication pair. The path number of the communication pair relies on the scale of bouncing switches at core layers. Most rack-to-rack DCNs such as Jellyfish use a random graph to achieve large-scale connection but do not use an efficient routing scheme: instead, they usually choose an appropriate general routing protocol that can work with their schemes. However, as mentioned in the previous subsection, it is unlikely that general routing protocols can scale to several thousands of routers in a DCN. Thus, routing is a significant problem for rack-to-rack DCNs, as described in Section 3.3.

Table 4 also highlights the common diameter (the longest path length between any two nodes in a DCN). A small diameter typically results in efficient routing [34]. In this respect, VL2 and PortLand usually follow 3-layer hierarchic model. The distance between any two nodes could not exceed 6 hops. On the other hand, the diameters in BCube and DCell rely on the level number of the cell. For example, the diameter of BCube$k$ network is $k+1$. Finally, Jellyfish is not a scale-free network such that the distance between any two nodes do not have an upper-bound.

Although a DCN-specific routing scheme benefits from reliability and efficiency compared to general-purpose routing protocols, they are limited in their ability to forward cloud applications to the Internet. In particular, the DCN-specific routing scheme does not provide an appropriate interface to the existing BGP from the Internet. If two DCNs

**Table 4**
Routing characteristics in the DCN architectures.

| DCN architectures | Routing characteristics | Common diameter | Maximum forwarding table size | Switch-default routing protocol |
|---|---|---|---|---|
| PortLand [30] | LDP to determine the switch position and the hop distances | 6 hops | 32 KB | ECMP style hashing to choose available forwarding path and match the flow |
| VL2 [31] | VLB to distribute traffic across a set of intermediate nodes | 6 hops | 16 KB | ECMP to distribute across equal-cost paths |
| DCell [34] | Divide-and-conquer algorithm to find the sub-path, DFR to find the whole path | $2^{k+1}-1$ hops ($DCell_k$) | 2 KB | shortest-path routing applying to DCell |
| BCube [35] | BSR to probe and calculate path, BFS to find parallel path | k+1 hops ($BCube_k$) | 2 KB | No use case for general routing protocol |
| Jellyfish [33] | No DCN-specific routing architecture | $\infty$ | – | k-shortest path applying to Jellyfish |

communicate with each other using different routing architectures, BGP faces a challenge to resolve inter-AS traffic. Some DCN-specific routing schemes implement a centralized path finding approach. For example, the LDP of PortLand (shown in Table 4) employs a fabric manager to compute the topology and send messages to every PortLand switch. However, the centralized routing approach is harder to when the DCN topology involves a large number of servers. Finally, some DCN-specific routing schemes (e.g., DFR and BSR in Table 4) compute the routing path locally. These approaches lack the global view of the whole topology such that the routing can cause loop problems at layers 2 and 3. To this end, the hybrid routing approach (e.g., SDN) may be useful, where some of the routing is handled by the centralized control plane while each switch only keeps the usual forwarding table based on the DCN topology.

### 6.2. Routing scheme for VI

A cloud computing network consists of different VIs that demand the routing of VI elements in an efficient way. Thus, there is a need to provide a routing scheme for VIs. In particular, the routing schemes can be performed either for a virtual network or a VM. Their features and cloud computing functionalities are as follows.

#### 6.2.1. Routing scheme for virtual network

With NV techniques, a routing scheme for virtual networks is designed using special tunneling mechanisms such as MAC-in-IP or MAC-in-MAC. Since most of these routing schemes use full IP functionality and independence from the device's location, they reduce the amount of layer 2 broadcast that is needed.

There are two main configurations that can achieve the tunnel mechanism. One is to design the self-learning algorithm in order to learn the VI topology via broadcast or multicast. In this respect, IETF RFC 5556 defines Transparent Interconnect of Lots of Links (TRILL) [86]. TRILL resolves multiple path diversity in an Ethernet. The control plane of TRILL uses the IS-IS protocol to determine the VI topology. When a packet is broadcast to all the TRILL nodes, the edge nodes interpret and de-encapsulate the MAC address from the packet header. Likewise, IEEE 802.1aq defines Shortest Path Bridging (SPB) [87], which resolves the bandwidth-wasting problem inherent in multi-path communication. SPB employs MAC-in-MAC tunneling. Each SPB node in the area will self-learn three kinds of trees: a shortest path tree for unicast and multicast, an equal-cost tree for load balance, and a multicast tree for unknown packets.

The other approach is to use static routing, in which the virtual network information is pre-configured at each node. In this respect, SPAIN in [88] can calculate multiple VLANs and pre-configure these results in all SPAIN areas. SPAIN also merges the redundant paths into multiple VLANs that can be used by multiple disjoint sub-trees. Two nodes can select different VLANs according to a wire's bandwidth. As a result, even a node trying to flood a link can use different VLANs to do it. Because SPAIN only requires MAC address learning and VLAN support, the commodity switch can deploy the protocol with fewer modifications.

Most schemes achieve an overlay routing for virtual networks within any network topology. However, both self-learning and pre-configuration approaches lack scalability in routing cloud service in a DCN. In particular, for a self-learning routing scheme, its learning abilities are limited to a small-topology network. However, since a virtual network originating from a tenant is usually complex, due to the need to connect thousands of VMs, the switch might cause large overhead in topology learning. On the other hand, the pre-configuration approach is only available for a small range of virtual networks because the forwarding table entries of each switch have a limited size, and massive amounts of virtual network information would exhaust the table capacity.

#### 6.2.2. Routing scheme for VM

In a cloud DCN, VMs share the physical NIC in the host server, and so each of them have isolated IP and MAC addresses. This means that a cloud DCN must route more end-to-end communications than a DCN without virtualization technique. Therefore, designing an efficient source routing scheme for VM traffic is needed for a cloud DCN. To this end, NetLord in Ref. [89] changes per-VM routing into per-host routing, which follows encapsulation with MAC-in-(IP+MAC). In each server, an agent is installed to manage the packet encapsulation for all VMs within the server. NetLord uses SPAIN to broadcast its topology at layer 2. The egress switch forwards a packet to its destination ingress switch according to its layer 2 lookup. When the ingress switch

receives the packet, it forwards it to its destination server according to the extra IP header.

Encapsulating VM information in the physical infrastructure can simplify cloud DCN routing and should solve the following challenges. First, the encapsulated agent, such as an end host or router, should allocate additional memory buffer in the respective equipment to record the encapsulation information. This will occupy an additional buffer that could be used for other computational activities, such as forwarding or I/O control. Second, since VM information becomes the payload encapsulated in the packet, the packet header should be extended for this payload so that each router can interpret it and make appropriate decisions whether to forward or not. This header extension increases the computational burden on the whole networks switches. Finally, such a routing scheme requires the modification of each end host kernel or the logic module of all the switches in the DCN. Modifications at this scale are unlikely to be deployed for a large-scale DCN.

### 6.3. Open research issues for cloud computing

Since a DCN has more volume and connective complexity than any other Internet cluster, and because DCN topologies vary greatly, most of the existing routing schemes are affected by the two aforementioned factors in which routing QoS incurs high latency. In particular, the following aspects of routing should be optimized in the future in order to meet scalability requirements.

- Forwarding table size is the most strongly affected when scaling up a cloud DCN, especially when a cloud DCN switch installs source routing protocols. Since today's commodity switches have limited forwarding table size as shown in Table 4, configuring or learning thousands of neighbors or information about the next hops is difficult and requires frequent deleting of the oldest entries. Although most DCN architectures have begun to minimize forwarding table sizes by grouping of DCN switches, e.g., by recording per-pod entries in the Fat-tree switch, they still do not adequately reduce the switch record volume.
- The complexity of DCN topology affects the self-learning in the layer 2 switch, as described in Section 6.2.2. The key solutions to this in the future will include reducing the number of hops from the root, avoiding path loops, and more efficient MAC addressing. To this end, a DCN topology should shorten the diameter and reduce redundant paths (see Section 3.4). On the other hand, there is a need to have a heuristic algorithm that could replace existing algorithms such as STP or LDP, which coordinate efficient information exchange between nodes. This requires an application of graph theory in the control plane.
- The burst traffic of a DCN will increase the lookup latency of each switch. That is because each switch has limited memory size to duplicate the packet and determine the forwarding. If a switch's memory is full due to excessive duplication, the incoming packet will be dropped. This is commonly seen in the core layer of a three-tier architecture. To address this problem, the source routing should consider load balancing based on knowledge of congested paths. This technique requires a fast and accurate congestion-finding algorithm.
- Many cloud applications, such as big data applications, require fault tolerant routing in the DCN, in which the DCN uses either a fault localization algorithm or redundant paths (e.g., ECMP). However, fault localization routing usually creates the aforementioned challenges of layer 2 self-learning, because a redundant path can create multicast loop-back and loop-forwarding problems. Further research in routing design is therefore needed.

## 7. Conclusion

Cloud computing provides a large-scale and crucial shared resource for massive amounts of users worldwide, creating a focal point for research and development efforts. DCN, which attempts to build connection between servers, is an efficient framework for cloud computing. In this paper, we outlined existing techniques of DCN for cloud computing. In particular, we classified key challenges and opportunities along the cloud computing architecture, which covers the physical DCN architecture, virtualized infrastructure, and DCN routing.

We provide a deep analysis of the features of existing DCN architectures, including electrical, optical, SDN, and wireless DCNs, and their feasibilities for use in cloud infrastructure. Although there is no precise evaluation of the cloud computing performance of different DCN architectures, this paper also poses a set of topology features that either help to estimate the cloud computing performance, or direct future DCN designs. We believe that the cloud-specific DCN, which have not yet appeared, will be implemented in the future if they address the respective challenges.

This paper presents the challenges of VI and deeply analyzes the future design of DCNs for VI. In particular, in order to provide on-demand resource flexibility, the existing VDC algorithms require placing the VM in a dynamic way. Likewise, the existing virtual networks should consider proper isolation between tenants. In the future, the container and application function emerge as essential elements of cloud activities. They are also very demanding and incur challenges to the existing DCN architectures.

Also, the issues regarding routing have been discussed and concluded. Most of the investigated solutions can be classified by being centralized and distributed approaches, DCN-specific and general approaches, hybrid solutions, etc. We show that the DCN scalability is the most constraining problem affecting the routing, which can be addressed in the future by the collaboration of different DCN elements.

In actuality, all new activities in the cloud create novel opportunities and challenges for the cloud data center, while solving certain traditional problems. Most of the articles presented in this survey attempted to solve these difficulties in managing both physical and virtual architectures. Additionally, interaction among multiple areas may provide innovative solutions for network-cloud convergence, which will significantly enhance the performance of the next generations cloud infrastructure. We hope that this paper will be helpful in suggesting the research road ahead, in order to allow the improvement of DCN-related techniques for cloud service.

## Acknowledgements

This work was supported by National Science and Technology Major Project (No. 2013ZX03002004), NRF Singapore CREATE Program E2S2, the Shanghai Science and Technology Development Fund for High-Tech Achievement Translation under Grant No. 14511100902, National Natural Science Foundation of China (Grant No. 61202374) and Shanghai Key Laboratory of Scalable Computing and Systems.

## References

[1] Cloud Computing Market, Global Cloud ComputingMarket:270 Billion By 2020, 2015, http://www.cloudcomputingmarket.com/.
[2] K. Chen, C. Hu, X. Zhang, K. Zheng, Y. Chen, A.V. Vasilakos, Routing in data centers: insights and future directions, IEEE Netw Mag 25 (2011) 6–10.
[3] D. Abts, B. Felderman, A guided tour through data-center networking, Queue 10 (5) (2012) 10:10–10:23.
[4] M. Chen, H. Jin, Y. Wen, V.C. Leung, Enabling technologies for future data center networking: a primer, Netw IEEE 27 (4) (2013) 8–15.
[5] A. Wang, M. Iyer, R. Dutta, G.N. Rouskas, I. Baldine, Network virtualization: technologies, perspectives, and frontiers, J Lightwave Technol 31 (4) (2013) 523–537.
[6] M. Chen, S. Mao, Y. Zhang, V. Leung, Big data: related technologies, challenges and future prospects, Springer briefs in computer science,, Springer, 2014.
[7] D. Kliazovich, P. Bouvry, S.U. Khan, Greencloud: a packet-level simulator of energy-aware cloud computing data centers, J Supercomput 62 (3) (2012) 1263–1283.
[8] K. Bilal, S.U.R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, et al., A taxonomy and survey on green data center networks, Future Gener Comput Syst 36 (2014) 189–208.
[9] K. Bilal, S. Khan, A. Zomaya, Green data center networks: challenges and opportunities, in: Proceedings of 11th international conference on frontiers of information technology (FIT), 2013, 2013, pp. 229–234.
[10] S. Sen, D. Shue, S. Ihm, M.J. Freedman, Scalable, optimal flow routing in datacenters via local link balancing, in: Proceedings of the ninth acm conference on emerging networking experiments and technologies, CoNEXT '13, 2013, pp. 151–162.
[11] T.G. Griffin, G. Wilfong, An analysis of BGP convergence properties, in: ACM SIGCOMM computer communication review, 29, ACM, 1999, pp. 277–288.
[12] C. Guo, G. Lu, H.J. Wang, S. Yang, C. Kong, P. Sun, et al., Secondnet: a data center network virtualization architecture with bandwidth guarantees, in: Proceedings of the 6th international conference, in: Co-NEXT '10, ACM, New York, NY, USA, 2010, pp. 15:1–15:12.
[13] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, et al., Data center network virtualization: a survey, Commun Surv Tutor IEEE 15 (2) (2013) 909–928.
[14] NIST, National Institute of Standards and Technology (NIST), 2010-2015, (http://www.nist.gov/).
[15] D. Borgetto, M. Maurer, G. Da-Costa, J.-M. Pierson, I. Brandic, Energy-efficient and SLA-aware management of IaaSS clouds, in: Proceedings of the e-Energy, 2012.
[16] S. Walraven, E. Truyen, W. Joosen, Comparing PaaS offerings in light of SaaS development, Computing 96 (8) (2014).
[17] H. Ballani, K. Jang, T. Karagiannis, C. Kim, D. Gunawardena, G. OShea, Chatty tenants and the cloud network sharing problem, in: Proceedings of the 10th USENIX conference on networked systems design and implementation, NSDI, 2013.
[18] M. Kajko-Mattsson, SLA management process model, in: Proceedings of the 2nd international conference on interaction sciences: information technology, culture and human, ICIS '09, ACM, New York, NY, USA, 2009, pp. 240–249.
[19] M. Chen, S. Mao, Y. Liu, Big data: a survey, 19, Mobile Netw Appl, 2014, pp. 171–209.
[20] Yahoo, Hadoop, http://developer.yahoo.com/hadoop/.
[21] M. Washam, Automating microsoft azure infrastructure services: from the data center to the cloud with powershell, 1st, O'Reilly Media, Inc., 2014.
[22] M. Burke, R.E. Kraut, Growing closer on Facebook: changes in tie strength through social network site use, in: Proceedings of the 32Nd annual ACM conference on human factors in computing systems, CHI '14, 2014, pp. 4187–4196.
[23] D. Zats, T. Das, P. Mohan, D. Borthakur, R. Katz, Detail: reducing the flow completion time tail in datacenter networks, ACM SIGCOMM Comput Commun Rev 42 (4) (2012).
[24] J. Dean, S. Ghemawat, Mapreduce: Simplified data processing on large clusters, Commun ACM 51 (1) (2008) 107–113.
[25] M. Alizadeh, A. Greenberg, D.A. Maltz, J. Padhye, P. Patel, B. Prabhakar, et al., Data center TCP (DCTCP), in: Proceedings of the ACM SIGCOMM 2010 Conference, in: SIGCOMM '10, ACM, New York, NY, USA, 2010, pp. 63–74.
[26] T. Benson, A. Akella, D.A. Maltz, Network traffic characteristics of data centers in the wild, in: Proceedings of the 10th ACM SIGCOMM conference on internet measurement, IMC '10, ACM, New York, NY, USA, 2010, pp. 267–280.
[27] K. Chen, A. Singlay, A. Singhz, K. Ramachandranz, L. Xuz, Y. Zhangz, X. Wen, Y. Chen, Osa: An optical switching architecture for data center networks with unprecedented flexibility, in: Proceedings of the 9th USENIX conference on networked systems design and implementation, NSDI'12, USENIX Association, Berkeley, CA, USA, 2012, p. 18.
[28] M. Chowdhury, I. Stoica, Coflow: a networking abstraction for cluster applications, in: Proceedings of the 11th ACM workshop on hot topics in networks, HotNets-XI, 2012, pp. 31–36.
[29] M. Al-Fares, A. Loukissas, A. Vahdat, A scalable, commodity data center network architecture, ACM SIGCOMM Comput Commun Rev 38 (4) (2008).
[30] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, A. Vahdat, Portland: A scalable fault-tolerant layer 2 data center network fabric, in: Proceedings of the ACM SIGCOMM 2009 conference on data communication, SIGCOMM '09, ACM, New York, NY, USA, 2009, pp. 39–50.
[31] A. Greenberg, J.R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, et al., Vl2: a scalable and flexible data center network, Commun ACM 54 (3) (2011).
[32] G. Wang, D.G. Andersen, M. Kaminsky, K. Papagiannaki, T.E. Ng, M. Kozuch, M. Ryan, c-through: part-time optics in data centers, in: Proceedings of the ACM SIGCOMM 2010 conference, in: SIGCOMM '10, ACM, New York, NY, USA, 2010, pp. 327–338.
[33] A. Singla, C.-Y. Hong, L. Popa, P.B. Godfrey, Jellyfish: Networking data centers randomly, in: Proceedings of the 9th USENIX conference on networked systems design and implementation, NSDI'12, Berkeley, CA, USA, 2012, p. 17.
[34] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, S. Lu, Dcell: a scalable and fault-tolerant network structure for data centers, in: Proceedings of the ACM SIGCOMM 2008 conference on data communication, in: SIGCOMM'08, ACM, New York, NY, USA, 2008, pp. 75–86.
[35] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, et al., Bcube: A high performance, server-centric network architecture for modular data centers, in: Proceedings of the ACM SIGCOMM 2009 Conference on data communication, in: SIGCOMM'09, ACM, New York, NY, USA, 2009, pp. 63–74.
[36] L. Gyarmati, T.A. Trinh, Scafida: a scale-free network inspired data center architecture, ACM SIGCOMM Comput Commun Rev 40 (5) (2010) 4–12.
[37] K. Bilal, S.U. Khan, L. Zhang, H. Li, K. Hayat, S.A. Madani, et al., Quantitative comparisons of the state-of-the-art data center architectures, Concurrency and Computation: Practice and Experience 25 (12) (2013) 1771–1783.
[38] K. Bilal, M. Manzano, S. Khan, E. Calle, K. Li, A. Zomaya, On the characterization of the structural robustness of data center networks, IEEE Trans Cloud Comput 1 (1) (2013) 1.
[39] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, Openflow: enabling innovation in campus networks, Comput Commun Rev 38 (2) (2008) 69–74.
[40] A.V. Vasilakos, L. Zhe, S. Gwendal, Y. Wei, Information centric network: research challenges and opportunities, J Netw Comput Appl 52 (2015) 1–10.
[41] T. Sahar, A. Furqan, K. Iyad, K. Mohamed, S. Reda, N.R. George, Spectrum management techniques for elastic optical networks: a survey, Opt Switch Netw 13 (2) (2014) 34–48.
[42] W. Shieh, C. Athaudage, Coherent optical orthogonal frequency division multiplexing, Electron Lett 42 (10) (2006) 587–589.
[43] N. Farrington, G. Porter, S. Radhakrishnan, H.H. Bazzaz, V. Subramanya, Y. Fainman, G. Papen, A. Vahdat, Helios: a hybrid electrical/optical switch architecture for modular data centers, in: Proceedings of the ACM SIGCOMM 2010 conference, in: SIGCOMM '10, 2010, pp. 339–350.
[44] C. Kachris, I. Tomkos, A survey on optical interconnects for data centers, Commun Surv Tutor IEEE 14 (4) (2012) 1021–1036.
[45] M. Chen, H. Jin, Y. Wen, V.C. Leung, Enabling technologies for future data center networking: a primer, Netw IEEE 27 (4) (2013).

[46] N.K. Dutta, Q. Wang, Semiconductor optical amplifiers, World scientific, 2013.
[47] W. Hui, N.R. George, Hierarchical traffic grooming: a tutorial, Comput Netw 69 (2014) 147–156.
[48] N.J. Ansari, S. Nema, R. Singh, A. Goel, Dense wavelength-division multiplexing networks waves system for high data rate communication, in: Proceedings of the international conference and workshop on emerging trends in technology, ICWET '10, 2010, pp. 443–444.
[49] J. Ning, T.-S. Kim, S.V. Krishnamurthy, C. Cordeiro, Directional neighbor discovery in 60 ghz indoor wireless networks, in: Proceedings of the 12th ACM international conference on modeling, analysis and simulation of wireless and mobile systems, MSWiM '09, 2009, pp. 365–373.
[50] X. Zhou, Z. Zhang, Y. Zhu, Y. Li, S. Kumar, A. Vahdat, et al., Mirror mirror on the ceiling: flexible wireless links for data centers, ACM SIGCOMM Comput Commun Rev 42 (4) (2012).
[51] Volkan, On the OpenFlow controllers VLAN article, 2012, (http://vlkan.com/blog/post/).
[52] B. Nunes, M. Mendonca, X. Nguyen, K. Obraczka, T. Turletti, A survey of software-defined networking: past, present, and future of programmable networks, Commun Surv Tutor IEEE (99) (2014) 1–18.
[53] J. Manar, S. Taranpreet, S. Abdallah, A. Rasool, L. Yiming, Software defined networking: state of the art and research challenges, Comput Netw 72 (2014) 74–98.
[54] H. Akram, G. Aniruddha, B. Pascal, S. Douglas C., G. Thierry, Software-defined networking: challenges and research opportunities for future internet, Comput Netw 75 (2014) 453–471.
[55] O. N.Foundation, Transforming networking with software-defined networking (SDN), 2015, (http://www.opennetworking.org/).
[56] L. Yang, R. Dantu, T. Anderson, R. Gopal, RFC 3746, April, 2004.Technical Report
[57] Cisco, Cisco system, Inc, 2013, (http://www.cisco.com/).
[58] OpenDaylight organization, OpenDaylight: a Linux fundation collaborate project, 2015, (http://www.opendaylight.org/).
[59] Juniper, Contrail, 1999-2015, (http://www.juniper.net/).
[60] Hatem Naguib, VMware NSX network virtualization, 2013, (http://blogs.vmware.com/vmware/2013/03/).
[61] H.H. Liu, X. Wu, M. Zhang, L. Yuan, R. Wattenhofer, D. Maltz, Zupdate: updating data center networks with zero loss, ACM SIGCOMM Comput Commun Rev 43 (4) (2013).
[62] Y. Dong, Y. Chen, Z. Pan, J. Dai, Y. Jiang, Renic: architectural extension to sr-iov i/o virtualization for efficient replication, ACM Trans Archit Code Optim 8 (4) (2012).
[63] S. Zhou, W. Cai, B.-S. Lee, S.J. Turner, Time-space consistency in large-scale distributed virtual environments, ACM Trans Model Comput Simul 14 (1) (2004).
[64] Docker, Build, ship and run any app, anywhere, 2015, (https://www.docker.com/).
[65] L. Rizzo, G. Lettieri, Vale, a switched ethernet for virtual machines, in: Proceedings of the 8th international conference on emerging networking experiments and technologies, CoNEXT '12, 2012, pp. 61–72.
[66] D. Crisan, R. Birke, G. Cressier, C. Minkenberg, M. Gusat, Got loss? get zovn!, ACM SIGCOMM Comput Commun Rev 43 (4) (2013) 423–434.
[67] K. Onoue, N. Matsuoka, J. Tanaka, Host-based multi-tenant technology for scalable data center networks, in: Proceedings of the eighth ACM/IEEE symposium on architectures for networking and communications systems, ANCS '12, 2012, pp. 87–98.
[68] M. Blaze, A.D. Keromytis, Trust management for IPsec, ACM Trans Inf Syst Secur 5 (2) (2002) 95–118.
[69] P.J. Frantz, G.O. Thompson, Vlan frame format, 1999. US Patent 5,959,990.
[70] VMWare, VXLAN: solving the datacenter network challenge, 2015, (http://www.vmware.com/solutions/).
[71] Microsoft, NVGRE, VXLAN and what Microsoft is doing right, 2011, (http://networkheresy.com/2011/10/03/).
[72] E. Gamess, R. Surós, An upper bound model for TCP and UDP throughput in ipv4 and ipv6, J Netw Comput Appl 31 (4) (2008).
[73] H. Ballani, P. Costa, T. Karagiannis, A. Rowstron, Towards predictable datacenter networks, in: Proceedings of the ACM SIGCOMM 2011 conference, in: SIGCOMM '11, ACM, New York, NY, USA, 2011, pp. 242–253.
[74] D. Xie, N. Ding, Y.C. Hu, R. Kompella, The only constant is change: incorporating time-varying network reservations in data centers, ACM SIGCOMM Comput Commun Rev 42 (4) (2012) 199–201.
[75] L. Popa, A. Krishnamurthy, S. Ratnasamy, I. Stoica, Faircloud: sharing the network in cloud computing, in: Proceedings of the 10th ACM workshop on hot topics in networks, HotNets-X, ACM, New York, NY, USA, 2011, pp. 22:1–22:6.
[76] J. Guo, F. Liu, D. Zeng, J. Lui, H. Jin, A cooperative game based allocation for sharing data center networks, in: Proceedings of the IEEE INFOCOM, 2013, IEEE, 2013, pp. 2139–2147.
[77] L. Popa, P. Yalagandula, S. Banerjee, J.C. Mogul, Y. Turner, J.R. Santos, Elasticswitch: practical work-conserving bandwidth guarantees for cloud computing, ACM SIGCOMM Comput Commun Rev 43 (4) (2013) 351–362.
[78] K. Ren, C. Wang, Q. Wang, Security challenges for the public cloud, Int Comput IEEE 16 (1) (2012) 69–73.
[79] L. Vanbever, S. Vissicchio, C. Pelsser, P. Francois, O. Bonaventure, Seamless network-wide IGP migrations, in: Proceedings of the ACM SIGCOMM 2011 conference, in: SIGCOMM '11, 2011, pp. 314–325.
[80] IETF, RFC 1142, 1990, (http://tools.ietf.org/html/rfc1142).
[81] R. Rastogi, Y. Breitbart, M. Garofalakis, A. Kumar, Optimal configuration of OSPF aggregates, IEEE/ACM Trans Netw 11 (2) (2003).
[82] X. Xiao, A. Hannan, B. Bailey, L.M. Ni, Traffic engineering with MPLS in the internet, Netw IEEE 14 (2) (2000) 28–33.
[83] A. Iselt, A. Kirstadter, A. Pardigon, T. Schwabe, Resilient routing using MPLS and ECMP, in: Proceedings of 2004 workshop on high performance switching and routing, 2004. HPSR., IEEE, 2004, pp. 345–349.
[84] B. Awerbuch, I. Cidon, S. Kutten, Optimal maintenance of a spanning tree, J ACM 55 (4) (2008).
[85] X. Zhang, F. Zhou, X. Zhu, H. Sun, A. Perrig, A.V. Vasilakos, H. Guan, DFL: Secure and practical fault localization for datacenter networks, IEEE/ACM Trans Netw 22 (4) (2014) 1218–1231.
[86] R. Perlman, D. Eastlake, D. Dutt, S. Gai, A. Ghanwani, Routing bridges (rbridges): base protocol specification, 2015, (http://tools.ietf.org/html/rfc6325).
[87] IEEE, 802.1aq, 2013, (http://etherealmind.com/).
[88] J. Mudigonda, P. Yalagandula, M. Al-Fares, J.C. Mogul, Spain: cots data-center ethernet for multipathing over arbitrary topologies, in: Proceedings of the 7th USENIX conference on networked systems design and implementation, NSDI'10, 2010, p. 18.
[89] J. Mudigonda, P. Yalagandula, J. Mogul, B. Stiekes, Y. Pouffary, Netlord: a scalable multi-tenant network architecture for virtualized datacenters, in: Proceedings of the ACM SIGCOMM 2011 conference, in: SIGCOMM '11, 2011, pp. 62–73.
[90] Y. Liu, Y. Li, Y. Wang, A.V. Vasilakos, J. Yuan, Achieving efficient and fast update for multiple flows in software-defined networks, DCC@SIGCOMM (2014) 77–82.
[91] F. Xu, F. Liu, H. Jin, A.V. Vasilakos, Managing performance overhead of virtual machines in cloud computing: A survey, state of the art, and future directions, Proceedings of the IEEE 102 (1) (2014) 11–31.
[92] G. Wei, A.V. Vasilakos, Y. Zheng, N. Xiong, A game-theoretic method of fair resource allocation for cloud computing services, The Journal of Supercomputing 54 (2) (2010) 252–269.
[93] D. Xu, X. Liu, A.V. Vasilakos, Traffic-aware resource provisioning for distributed clouds, IEEE Cloud Computing 2 (1) (2015) 30–39.

**Bin Wang** is currently a Ph.D. student of Computer Science and Engineering at Shanghai Jiao Tong University (China). He received the M.S. degree of Software Engineering from Xi'an Jiao Tong University (China) in 2009. In 2005 he received his B.S. degree at School of Mechanical and Electrical Engineering from Xidian University (China). His research interests include virtual machine, data center network and cloud computing.

**Zhengwei Qi** is an assistant professor in the School of Software at the Shanghai Jiao Tong University (China). He received his B.Eng. and M.Eng. degrees from Northwestern Polytechnical University in 1995 and 1999, respectively. He received his Ph.D. in Computer Science and Engineering from Shanghai Jiao Tong University in 2005. His research interests are distributed computing, virtualized security, model checking, program analysis and embedded systems.

**Ruhui Ma** received his Ph.D. degree in computer science from Shanghai Jiao Tong University, China, in 2011. He has worked as a postdoc at Shanghai Jiao Tong University (2012 and 2013), at McGill University, Canada (2014), respectively. He is an Assistant Professor in the Department of Computer Science and Engineering at SJTU. His main research interests are in virtual machines, computer architecture and network virtualization.

**Haibing Guan** now is the vice dean of School of Electronic, Information and Electronic Engineering, Shanghai Jiao Tong University, also in charge of the Ministry-province jointly-constructed cultivation base for state key lab and the Shanghai Key Laboratory of Scalable Computing and Systems. He obtained his Ph.D degree from Tongji University in 1999 and worked in Shanghai Jiao Tong University since 2002. His major research interests include computer system , Cloud Computing. He hosted several national and ministry-province joint programs in network security, virtualization technology, network storage, data communication, etc. Research works of Prof. Guan are funded by Ministry of Science and Technology, National Natural Science Foundation, Ministry of Education and Shanghai government many times. Moreover, he keeps touch with institutes from the industry like Research Laboratory of IBM, Intel, Microsoft, HP and native large enterprises.

**Athanasios V. Vasilakos** is the professor at Dept of Computer Science, Electrical and Space Engineering, Lulea University of Technology, Lulea, Sweden. He has served as General Chair, and Technical Program Committee Chair for many internation conferences. He also served or is serving as Editor or/and Guest Editor for many technical journals, such as IEEE TNSM, IEEE TSMC-partB, IEEE TITB, IEEE JSAC special issues of May 2009, Jan. 2011, March 2011, ACM TAAS and IEEE Communications Magazine. He is founding Editor-in-Chief of the International Journal of Adaptive and Autonomous Communications Systems (IJAACS) and the International Journal of Arts and Technology (IJART). He is also General Chair of the Council of Computing and Communications of the European Alliances for Innovation.