Autonomous Network and IT Resource Management for Geographically Distributed Data Centers

Yiwen Shen, Payman Samadi, and Keren Bergman

Abstract—As enterprises struggle to meet the demand for elastic cloud computing services in distributed data centers (DCs), a significant portion of the total expenditure is spent on the procurement and maintenance of server and network equipment. Network and server resource utilization efficiency is therefore crucial to minimize overall costs and increase return on investment. We present an adaptive and synergistic management of both network and IT resources with an autonomous control plane on a converged inter/intra-DC network to address this issue. The control plane uses its awareness of both network and IT resource usage to provide dynamic resource allocations to meet quality of service guarantees, while also allowing for reduction in operational expenses by consolidating virtual machines to maintain high CPU usage on the active servers. We built a three-node inter-DC experimental testbed to demonstrate the feasibility of our concept and show that our algorithm successfully provisions both network resources and consolidates IT resources under various load conditions.

Index Terms—Autonomous control plane; Data center networks; Networks; Network provisioning strategy; Quality of service (QoS); Software-defined networking (SDN); Traffic monitoring.

I. INTRODUCTION

Over the past few years, data center (DC) networks have become a crucial enabler of cloud-based services and virtualization. According to Gartner Inc., the current public cloud services market in 2017 totals $247 billion, and is expected to grow by more than 50% by 2020 [1]. This rapid growth has resulted in a significant increase in the number of servers within cloud DCs, with current facilities housing tens to hundreds of thousands of servers [2]. Because DC networks form the groundwork that supports this growth, contention and low utilization of shared network and compute resources in DC networks can be a major contributor to performance degradation. This is especially true for cloud computing services built on a virtualized infrastructure, where much of the compute resources are shared among multiple end users and can show significant load fluctuations due to user demand. Specifically, DC workload patterns have been found to have weekend/weekday variations [3] but exhibit burstiness and are generally unpredictable on shorter time scales [4,5]. To maximize utilization of the underlying physical infrastructure for unpredictable traffic behavior, an adaptive DC resource management strategy that provisions resources on demand is required. There are two types of targeted resources: network resources, primarily link bandwidth, and IT resources, including CPU, physical memory, and hard disk capacity. Network resources can be effectively managed at the flow level over the DC network, and IT resources are managed through allocation and migration of virtual machines (VMs) in a cloud DC.

Software-defined networking (SDN) can provide the necessary functionalities for managing both network and IT resources. An SDN controller collects network resource information and disseminates control plane intelligence to the physical network entities to adapt the network to current load conditions. This is enabled by the separation of the data and control planes, which allows centralized management of network components from the higher layers. With regular monitoring of network domain traffic behavior, the SDN controller can mitigate network congestion by intelligently allocating dynamic bandwidth resources to heavily subscribed links. Additional modifications to the SDN application allow the controller to obtain IT resource usage statistics of the servers in the DC and subsequently control VM placement to achieve high server utilization. The combination of these capabilities allows the SDN controller to perform IT resource management with awareness of the bandwidth usage of the compute elements and links in the network, creating an additional degree of control for network-wide performance optimization.

In this work, we built an autonomous and on-demand network and IT resource provisioning SDN control plane architecture for metro-scale DC networks. It combines both network and IT resource provisioning capabilities to support specific quality of service (QoS) requirements and optimizes resource utilization under rapid and dynamically changing load variations. At the same time, it aims to minimize cost and energy usage of the distributed DC computing elements by consolidating server workloads. This is achieved through i) second-scale monitoring of both the link bandwidth usage of the network and IT resources usage of the racks; ii) enabling connections with various QoSs through two types of links, namely, background and dynamic; iii) triggering autonomous live migrations of VMs.

Manuscript received July 17, 2017; revised November 21, 2017; accepted November 29, 2017; published January 31, 2018 (Doc. ID 302532).

The authors are with the Lightwave Research Laboratory, Department of Electrical Engineering, Columbia University, New York, New York 10024, USA (e-mail: ys2799@columbia.edu).

https://doi.org/10.1364/JOCN.10.00A225
operating in the servers to consolidate workload; and iv) managing network connections in both explicit operations (request by operator) and implicit operations (automated assignment based on traffic characteristics) by the SDN control plane. We built an experimental testbed of three DC nodes, two fully emulated and one in the control plane, and demonstrate its capability to provision both network and IT resources under several scenarios. Experimental results show that our algorithm is able to successfully consolidate VMs to obtain high server utilization as well as allocate the necessary network bandwidth on-the-fly for varying traffic demands. This work is an extension of a previous work involving the management of network bandwidth resources for a converged intra/inter-optical-DC network, but does not address the provisioning and management of IT resources [6]. For this current work we focus more on the dynamic provisioning capabilities of the control plane of both network and IT resources.

In Section II, we review the related works on DC network architectures that enable provisioning of bandwidth and IT resources on-demand. In Section III, we describe the SDN control plane and network architecture. Section IV presents the experimental results obtained from the testbed we built. Section V is a discussion on the application, flexibility, and obstacles faced by our proposed architecture, and Section VI presents our conclusion.

II. RELATED WORKS

There have been many works investigating dynamic provisioning of bandwidth in DC networks and server-resource-aware VM consolidation for managing IT resources. VM consolidation is the act of clustering VMs running on underutilized physical servers into fewer hosts to improve server resource utilization. There are two types of VM consolidation, namely, static and dynamic. Static consolidation uses historical resource utilization as input to predict future resource use trends in order to map VMs to physical hosts, whereupon the assignment remains unchanged for months [7]. Dynamic server consolidation is useful for a more unpredictable workload and is carried out on a shorter time scale. Examples of this include Refs. [8–10]. Reference [8] presents a framework for the placement of VMs to minimize network power reduction by turning off unused top-of-rack (ToR) switches while satisfying as many network requests as possible. The algorithm first tries to place source and destination VMs under the same parent ToR, which allows only the common parent ToR to be turned on to route the demand. If that is not possible, a minimum weight path is used between possible pairs of ToRs. Reference [9] demonstrates network-aware placement and migration of VMs based on cost of migration and the available bandwidth to minimize congested links. It features a SDN bandwidth monitoring module to calculate a normalized weighted sum of the available bandwidth on its path over the largest demand for that VM, so that the best destination host can be chosen. Reference [10] formulates VM consolidation into a stochastic bin packing problem given the resources available, to achieve the largest possible server number reduction.

Network bandwidth is a resource crucial to the performance of the majority of cloud-based services. There are a large number of strategies proposed for fine-grained network resource provisioning for DC networks. Reference [11] describes a flexible-grid inter-DC network architecture and investigates efficient support of cloud and big data applications. Reference [12] presents a flexible optical metro node architecture and discusses methods and challenges in providing dynamic transport services in a distributed DC environment. References [13,14] demonstrated optical burst/packet switched network architectures capable of setting up direct lightpaths between racks in different metro DCs, which reduces inter-DC latency and provides better resource usage compared to conventional networks. For example, Ref. [13] features a distributed metro-scale DC network with a hierarchical SDN control plane that virtualizes the network into slices to meet client requests and service provisioning. Each metro DC features a ring topology with a central master node that handles messages from the SDN controller, allowing for rack-to-rack interconnections to be dynamically reconfigured for both intra- and inter-DC communications.

In these works, different methods for adaptive provisioning IT or network resources are shown, but each focuses on only one type of resource, either IT or network bandwidth. Works involved in the management of IT resources use the current state of the network traffic to make decisions on provisioning VMs, but do not influence the network’s physical topology or network capacity with additional resources. On the other hand, works involved in provisioning dynamic bandwidth do not address the issues of oversusing server CPU or memory. We seek to create a comprehensive control plane that manages both types of resource in a synergistic manner.

III. NETWORK ARCHITECTURE

Our network architecture features a control plane over a converged inter/intra-DC network that provides inter-DC connectivity through the provisioning of background DC-to-DC and dynamic rack-to-rack and/or pod-to-pod connections. The control plane also autonomously initiates live VM migrations to maintain high server utilization.

A. Data Plane

The data plane consists of the inter/intra-DC network and the racks/pods of the DCs, as shown in Fig. 1. The electronic packet switched (EPS) network provides both inter- and intra-DC connectivity, a different approach from our previous work where an optical gateway provided the inter-DC connectivity [15]. The network architecture supports two types of connections: i) background connections (shown in red), which provide basic DC-to-DC connectivity and are present at all times, and ii) dynamic connections (shown in green), which provide direct rack/pod connections between DCs, which transmit through the DC gateway. The background connections are shared by several traffic flows from different ToR switches. As these flows share the total bandwidth of the link, their throughput is proportionately limited by the
number of flows sharing the link. Therefore, background connections are intended for short-lived and low data rate traffic flows, which have no strict requirement in terms of bandwidth or latency. On the other hand, dynamic connections are used to perform large data rate transfers between racks in different DCs. They are provisioned on-the-fly directly between racks or pods through the DC gateway and are used for applications with strict latency or large bandwidth requirements. For more details on the operations of the data plane network architecture, please refer to our previous work [16].

Algorithm 1: Network and IT Resource Implicit Control Algorithm

New connection request \( R \) from DC\(_S\) to DC\(_D\);
Monitor the active Background and Dynamic connections, and IT resource usage percentage;
if Traffic on Background \( B \) DC\(_S\) to DC\(_D\) \( \geq \) Threshold \( T_0 \) then
   if SD of traffic on \( B \) \( \leq \) 1 then
      Create new Background link from DC\(_S\) to DC\(_D\)
   else if SD of traffic on \( B \) \( > \) 1 then
      Create new Dynamic link from DC\(_S\) to DC\(_D\)
   end if
end if
if IT resource usage on Server \( U \) \( \geq \) Threshold \( T_1 \) or Dynamic link bandwidth usage on Server \( Q \) \( \geq \) Threshold \( T_2 \) then
   Initialize live VM migration from Server \( U \) to next available Server \( V \)
else if IT resources usage on Server \( V \) \( \geq \) Threshold \( T_3 \) and IT resources usage on Server \( U \) \(< \) Threshold \( T_1 \) or Dynamic link bandwidth usage on Server \( V \) \( \geq \) Threshold \( T_4 \) and Dynamic link bandwidth usage on Server \( U \) \(< \) Threshold \( T_5 \) then
   Initialize live VM migration from Server \( V \) back to Server \( U \)
end if

B. Control Plane

The SDN control plane (Fig. 2) contains three modules: i) the traffic and IT resource usage monitor, ii) the resource usage optimizer, and iii) the topology and VM placement manager. At initial startup, the SDN control plane establishes connections with both the OpenFlow enabled switches and the DC servers. At regular second-scale intervals, the SDN controller obtains current rack-to-rack traffic statistics from the OpenFlow switches, as well as IT resources usage of each server in the DCs. The resource usage optimizer uses this information to determine whether new background or dynamic connections are needed and whether currently operating VMs need to be consolidated. If yes, the resource usage optimizer invokes the topology and VM placement manager to modify the flow tables of the relevant ToR switches to establish new links, and/or initiates VM live migrations to increase or reduce CPU usage on certain servers. The details of the decision-making process for establishing dynamic links as well as VM migrations are as follows.

The algorithm for the implicit operation of provisioning bandwidth and IT resources is shown in Algorithm I. To start, the initialization of the network architecture establishes the inter-DC background connections necessary for basic connectivity, and the controller establishes connections to the servers. After that, periodic monitoring obtains the
throughput of the background connections and consumption of IT resources. Monitoring of the throughput is performed by obtaining the current and previous values of the number of bytes sent by each flow on the shared link and dividing this difference by the monitoring interval time.

The first half of Algorithm 1 describes the bandwidth allocation algorithm for the metro network. The SDN controller uses the throughput values calculated through periodic traffic monitoring of each flow to determine whether the traffic on background connections \( B \) between source and destination DCs (DC\textsubscript{S} and DC\textsubscript{D}) is higher than a specified threshold. If so, the controller will calculate the standard deviation (SD) of the throughput between the different flows in \( B \). If the SD \( \leq 1 \), a new background connection is established between DC\textsubscript{S} and DC\textsubscript{D}. If the SD \( > 1 \), then a new dynamic connection is created for the flow that generates the highest traffic. This allows a dynamic connection to be allocated only if the largest flow has a significantly higher throughput than the others. Otherwise, if the flows are sharing the bandwidth of the link with an almost equal demand, then allocating a new background connection is fairer to each flow.

The second half of Algorithm 1 describes the mechanism for autonomous VM consolidation. The concept of this algorithm is analogous to the implicit algorithm for bandwidth allocation discussed previously. In the beginning, requests to the DC are handled by VMs consolidated onto a small number of active servers. The SDN controller obtains the current IT resource usage of each server in the distributed DC network at regular intervals. When the workload in a server increases past a certain threshold, causing performance to degrade if left unchecked, the controller will initiate VM live migrations to the next available server to distribute the workload, which can be located within the same rack in the same metro DC, or a rack located in another metro DC, which will also prompt a dynamic link allocation if necessary. This allows the disruption of service caused by lack of network bandwidth for the VM migration to be reduced to a minimum. On the other hand, VM migrations can also be used to reduce traffic hotspots. If a dedicated dynamic link is insufficient to provide the bandwidth necessary to meet demands, the controller will perform VM migrations to a second server to relieve network congestion by allowing part of the traffic to be directed to the second server despite sufficient CPU and memory resources in the first server.

The last “else if” statement in Algorithm 1 describes the consolidation process, where if the second server \( V \) that the VM was migrated to is still active but the original server \( U \) where the VM came from is consuming IT resources or dynamic link bandwidth below a certain threshold, meaning that it is no longer consuming as much resources as previously, then this VM is migrated back to \( U \) to maintain high CPU usage of the original server \( U \). This process allows the second server \( V \) to be turned off to save operational costs and energy consumption.

Overall, our control plane ensures that both network resources and IT resources are available to the servers in the distributed DC network at all times to mitigate congestion and maintain high CPU usage, while minimizing the number of active computation nodes. In this way, our architecture eliminates the costs associated with hardware maintenance and energy usage from keeping unnecessary servers active.

IV. EXPERIMENTAL TESTBED AND RESULTS

Our testbed (Fig. 3) consists of three DC nodes, and each DC consists of four racks, each connected to one server. The servers are equipped with a 6-core Intel Xeon processor, 24 GB memory, and a dual-port 10G network interface card (NIC), running Centos 7. VMs on each server are created using the Centos VM Manager with 8 GB of hard disk and 8 GB of local memory usage each. IT resource monitoring was performed using the psutil Python library. The SDN controller sends messages to the servers over a secure shell (SSH) connection. Each server contains a script for running the psutil function that returns the current system-wide CPU utilization as a percentage, which is then returned to the SDN controller for processing. VM migrations were performed using the functions from the libvirt Python library. Once a decision has been made to migrate a VM, the SDN controller uses the source host and destination host IP addresses to initiate the VM live migration.

ToR switches are implemented using two Pica8 10G OpenFlow switches logically divided to eight bridges. Each ToR has a 10G port to the server and a 10G port to the EPS switch with a direct-attached copper cable to the gateway. The DC gateway used for dynamic connections is implemented using the Pica8 switches. In our previous work, the DC gateway was supported by an optical gateway implemented using optical space switches, wavelength selective switches, and DWDM mux/demuxes [15].

In our previous works [17,18] where wavelength contention was an issue for optical DC gateways, the controller also assigned priorities to each new dynamic connection at this stage, so that higher priority connections could push existing lower priority connections. A new dynamic connection can be assigned priority 0, representing a connection for bulk traffic that is tolerant of delays, or priority 1, representing a critical connection that sends delay-sensitive traffic. In such a case, the controller can force an active dynamic connection with priority 0 to move to a lower data rate or to a background connection.

---

\[^1\]In our previous works [17,18] where wavelength contention was an issue for optical DC gateways, the controller also assigned priorities to each new dynamic connection at this stage, so that higher priority connections could push existing lower priority connections. A new dynamic connection can be assigned priority 0, representing a connection for bulk traffic that is tolerant of delays, or priority 1, representing a critical connection that sends delay-sensitive traffic. In such a case, the controller can force an active dynamic connection with priority 0 to move to a lower data rate or to a background connection.

---
The SDN control plane is implemented using the SDN controller Ryu and is connected to the ToRs, DC gateways, and servers via 1 Gbps campus Internet. The controller communicates with the switches using the OpenFlow protocol, while the connections to the servers are through TCP socket connections. DCs 1 and 2 are fully emulated, while DC 3 is implemented only in the control plane to achieve a mesh topology.

We performed three sets of experiments (Fig. 4) with CPU usage as the limiting IT resource. The top three graphs show the CPU usage percentage over time of the relevant servers. The bottom three graphs show the throughput of the servers. The top and bottom graphs of each experiment take place simultaneously for the same experimental procedure.

In the first scenario [Fig. 4(a)], we focus on the behavior of the control plane under varying computational loads. New requests arrive at DC1, causing new VMs to be spun up in Server 1, reflected in increasing CPU usage over time. The CPU usage of Server 1 continues to increase over time as more VMs are consolidated onto it, until, at the 54 s mark (shown by the first dotted black line), the CPU usage grows above a maximum threshold of 80%. This increase triggers a VM migration from Server 1 to Server 5 in DC 2, resulting in a decrease in CPU usage of Server 1 while CPU usage increases for Server 5 due to the offset workload. We also observe the effect on the network, shown by the short burst of traffic generated by Server 1 (bottom graph). When the CPU usage of Server 1 has decreased to a sufficient level below 30% while the CPU usage of Server 5 remains at a moderate level of 40%, the controller initiates a migration for this VM to move back to Server 1 to consolidate the workload (marked by the second dotted black line). We observe that Server 5’s CPU usage decreases to 0, while Server 1’s CPU usage increases slightly, until the request is completed. The traffic created by the migration back to Server 1 is also shown in the throughput of Server 5 in the bottom graph.

The second experiment [Fig. 4(b)] focuses on the behavior of the control plane under varying traffic demands and demonstrates the allocation of dynamic links for each server in DC 1. The experiment begins with each server using the shared background link and consuming approximately 1 Gbps throughput each, resulting in a total of approximately 4 Gbps of bandwidth on the background link, below the threshold for extra bandwidth provisioning. At approximately 10 s into the experiment, the bandwidth used by Server 1 grows with SD > 1, causing the control plane to allocate a dynamic link to meet this demand. The same behavior is demonstrated for Server 2 at approximately 28 s, Server 3 at 56 s, and Server 4 at 79 s. The dedicated dynamic link allocated to each server allows the servers to send and receive at a much higher data rate of approximately 7.5 Gbps. Note that the throughput for the flows occupying the dedicated dynamic links can reach to near full capacity of 10 Gbps, but we purposely cap the throughput at 7.5 Gbps in order to not trigger VM migrations caused by high bandwidth usage, illustrated in the next experiment. We also observe that CPU usage caused by the VMs running low workload applications is unaffected by throughput changes and dynamic link allocations.

The third experiment [Fig. 4(c)] shows the effects on CPU usage and network throughput when the consolidation of VMs onto Server 1 leads to high network usage and how the control plane handles this situation. To begin, Server 1 is running VMs that consume a low level amount of CPU capacity and network bandwidth. At the 23 s mark, new requests cause the workload to increase to over 50% and the throughput to approximately 7.5 Gbps. The CPU and bandwidth utilization is not enough to cause a VM migration, but traffic demands cause a dynamic link to be allocated. As the CPU usage on Server 1 is still low enough for new VMs to be created, the bandwidth usage now grows to near full capacity (90%) of the 10G link, causing a VM migration from Server 1 in DC 1 to Server 5 in DC 2 at the time, marked by the second dotted line. This VM migration leads to a reduced bandwidth utilization of approximately 6 Gbps by Server 1 and approximately 4 Gbps by Server 5. Server 1 continues to use the dynamic link, while Server 5 uses the background connection, both of which are
uncongested. We note that the throughput of Server 1 remains high for approximately 1 s beyond the dotted line instead of dropping immediately due to the additional traffic generated by the VM migration at that time. Additionally, since the VM has migrated to a different physical server, there is a small delay in Server 5’s throughput due to the time required to reestablish connection to the VM.

We demonstrate the feasibility of our control plane algorithm by subjecting our control plane to handle three different scenarios involving dynamic load variations on CPU usage, as well as both background and dynamic link bandwidth usage. In each scenario, the results show that the control plane properly detects high resource utilization for both network bandwidth and CPU usage and performs the appropriate actions to allocate resources when the thresholds are reached. Therefore, we demonstrate that it is possible to build a reconfigurable and autonomous resource provisioning control plane using commodity electronic switches that can steer bandwidth and consolidate VMs for efficient resource utilization under different application load conditions.

V. DISCUSSION

Our proposed control plane is designed to manage IT and network resources for metro-scale networks with distributed small to mid-sized DCs. Leveraging SDN, we take advantage of its ability to monitor the entire network domain to centrally aggregate statistics and provision both network and IT resources in an approximate and globally optimal way. With dynamic links available to be used on demand, we provide a simple and effective method to utilize the spare network capacity of the DC to offload traffic from congested links. We build upon this foundation of a reconfigurable network with the additional capability of IT resource management through VM consolidation for cloud computing applications, using the same principles of monitoring and provisioning on demand as used for the network. This created a DC network system that maintains an adequate level of both IT and network resources at all times, and our algorithm and experimental results show that the provisioning of these two types of resources will not become a bottleneck to the other for providing overall performance and reliability.

While SDN provides reliable monitoring of infrastructure information and efficient provisioning of resources, it also generates extra overhead in the network from gathering the usage statistics of both the switching components and the compute elements in the DCs. This is a well-known challenge for SDN networks due to the centralized nature of the network architecture. A possible approach to address the issue can be the following: the network can be divided among multiple SDN controllers, where each controller will perform the same network bandwidth and IT resource allocation algorithm for the section of the network that it controls. A higher level controller or orchestrator that acts as a mediator for any coordination would be required between the local controllers.

VI. CONCLUSION

Our dynamic resource provisioning control plane facilitates network- and server-aware control to adapt and orchestrate autonomous allocation of network and server resources in a synergistic manner. We show that our network architecture supports background and dynamic connections for different levels of QoS to meet the various bandwidth and latency requirements of incoming network connections. It also provides load-sensitive server management through VM migrations, which operates alongside its network bandwidth provisioning capabilities. We built a testbed with emulated DCs to demonstrate the functionalities offered by our control plane and demonstrated its feasibility.

ACKNOWLEDGMENT

This work was supported in part by CIAN NSF ERC (EEC-0812072), NSF NetS (CNS-1423105), and DoE Advanced Scientific Computing Research (ASCR) under Turbo Project (DE-SC0015867). We would also like to thank AT&T, Calient Technologies, and Polatis for their generous donations to our testbed.

REFERENCES


Yiwen Shen was born in Shanghai, China, in 1992. He received his B.Sc. in electrical engineering from the University of Toronto, Ontario, Canada in 2014 and his M.Sc. in electrical engineering from Columbia University, New York, New York, in 2015. He is currently a Graduate Research Assistant with the Lightwave Research Laboratory at Columbia University where he is pursuing his Ph.D. His research interests focus on the development of optical software-defined networks for data-center and high-performance computing applications.

Payman Samadi is a Research Scientist at the Lightwave Research Laboratory of Columbia University, New York, New York. He received his Ph.D. degree in photonics from McGill University, Montreal, Québec, Canada, in 2012. He was formerly with Optiwave Systems Inc. as a product manager (2011–2013). He has co-authored more than 30 articles in peer-reviewed conferences and journals and is a senior member of IEEE. His current research interests include data center network architecture, software-defined networking, application of optical interconnects for computing platforms, and big data analytics.

Keren Bergman (S87-M93-SM07-F09) received the B.S. degree from Bucknell University, Lewisburg, Pennsylvania, in 1988, and the M.S. and Ph.D. degrees from the Massachusetts Institute of Technology, Cambridge, Massachusetts, in 1991 and 1994, respectively, all in electrical engineering. She is currently the Charles Batchelor Professor of Electrical Engineering at Columbia University, New York, New York, where she is also the Scientific Director of the Columbia Nano Initiative. Her current research interests include optical interconnection networks for high-performance embedded computing, optical data center networks, and silicon photonic systems-on-chip.