LB-MAP: Load-Balanced Middlebox Assignment in Policy-Driven Data Centers

Manar Alqarni, Alexander Ing, and Bin Tang
Department of Computer Science
California State University Dominguez Hills, Carson, CA 90747, USA
Email: {malqarni1,aing1}@toromail.csudh.edu, btang@csudh.edu

***Abstract*—Middleboxes (MBs), such as firewalls and load balancers, are playing an increasingly important role in cloud data centers for security or performance purposes. The recent introduction of Software Defined Network (SDN) and Network Function Virtualization (NFV) in cloud data centers has greatly facilitated the efficient management of software-based middleboxes in data center networks. In policy-driven cloud data centers, it requires that virtual machine (VM) traffic traverses a sequence of specified middleboxes in order to achieve security and performance guarantee. Much research has been done to study how to place middleboxes inside data centers for cost-efficient VM traffic traversal. However, not much research has focused on load balance of middleboxes. In this paper we study the Load-Balanced Middlebox Assignment Problem (LB-MAP), which minimizes the communication energy cost of VM pairs while satisfying their policy requirement as well as the capacity constraint of the switches that the middleboxes are placed upon. We show that LB-MAP is equivalent to the classic minimum cost flow problem (MCF), which can be solved optimally and efficiently. We also design a suite of efficient heuristic algorithms based on different criteria viz. VM-Based, MB-Based, and VM+MB-Based. Via extensive simulations, we show that all the heuristic algorithms perform close to the optimal minimum cost flow algorithm, while VM+MB-Based performs best among all the heuristic algorithms. To the best of our knowledge, this is the first work that addresses the energy cost minimization for VM communications as well as load-balancing for middleboxes in policy-driven data centers.**

Keywords – Virtual Machine Communication, Middle Box Management, Load-Balancing, Policy-Driven Data Centers

I. Introduction

Middleboxes, also known as "network appliances" or "network functions (NFs)", are intermediary computer networking devices that transform, inspect, filter, or otherwise manipulate network traffic for purposes other than packet forwarding. Middleboxes are widely deployed in enterprise networks such as data centers and play an increasingly important role in improving the networks' security (e.g., firewalls and intrusion detection systems (IDSs)), performance (e.g., load balancers), as well as reducing the bandwidth cost (e.g., WAN optimizers).

A recent study [32] shows that the number of middleboxes is on par with the number of routers in enterprise networks. Traditional middleboxes are diverse, stateful systems supporting narrow specialized network functions. As such, they are mainly proprietary and purpose-built hardware, which is typically closed and expensive. Their deployment and operation represent a significant part of the network capital and operational expenditure as well as costing large amount of space and power consumption. This greatly impedes the equipment upgrade and service addition in any networks, inducing the so-called *network ossification problem* [22].

Network Function Virtualization (NFV) has been recently proposed to alleviate this problem [11], [26]. NFV is a network virtualization technology that virtualizes middleboxes (or network functions) into building blocks that create communication services. It allows network operators and service providers to implement middleboxes in software instead of purpose-built hardware. This dramatically improves the management efficiency of middleboxes as they can be moved to or instantiate in various locations in the network without new equipment installation and network operator involvement. Software Defined Networking (SDN) [13], [2], being a complementary technology to NFV, further alleviates the network ossification problem by moving management functions out of the hardware and placing them in a centralized controller. Such centralized management, coupled with standardized protocol (e.g. OpenFlow [24]) between the controller and network functions, enables dynamic and flexible configuration and placement of middleboxes inside networks.

The major enabler and beneficiary of NFV and SDN are cloud data centers, for two reasons. First, virtualization has become the core building block of modern data centers. The hardware resources such as CPU cycles, memory, and bandwidth are divided into smaller isolated computing units, known as *virtual machines* (VMs), which can be rented to tenants in a pay-as-you-go manner. Thus data centers provide an ideal enabling platform to implement and experiment the concepts of NFV and SDN. Second, cloud data centers deploy a variety of middleboxes to protect, manage and improve the performance of the applications and services [19]. As a result, the introduction of NFV and SDN has greatly improved the middlebox management in cloud data centers [32], [30], [15].

In order to provide security and performance guarantee in data centers, *network policies* are established to demand network traffic to traverse a sequence of specified middleboxes. For example, each VM communicating pair in data center could go through an IDS and a load balancer in that order so that malicious traffic can be filtered then trusted traffic be diverted to avoid network congestion. In fact, due to the diverse user applications demands, network policies have become an inseparable part of the Service Level Agreement (SLA) of data

centers, which sets the expectations and commitments between the cloud users and cloud service providers. Satisfying those policies or not become an important measurement of the efficiency and efficacy of any data centers; we refer to such data centers as *policy-driven data centers*.

However, there exists a dilemma in the middlebox management of policy-driven data centers. On one side, the middleboxes have limited packet hardware resources such as CPU, memory, or accelerators. The SDN switches upon which middleboxes are installed use Ternary Content-Addressable Memory (TCAM), which is expensive in both cost as well as power consumption, making the amount of forwarding rules available at each switch very limited. On the other side, middleboxes involve complex and extensive processing to capture application-semantics by using deep packet inspection. As such, overload is one of the dominant reasons for middlebox failures [32], causing packet loss, traffic delay as well as wasting energy consumption. How to well-balance the workload of middleboxes while minimize the VM communication energy cost is an important problem in policy-driven data centers.

Fortunately, due to the software implementation of middleboxes brought by NFV and SDN, it is possible to replicate and place multiple copies of the same middlebox inside the policy-driven data centers for the purpose of load-balancing and fault tolerance [12]. Each copy is referred to as an *instance* of the middlebox. In this paper we consider that there are multiple instances of one type of middlebox in the policy-driven data center. To satisfy the policy requirement of security or performance, each VM communication pair must traverse one of the middlebox instances while different VM pairs can traverse different instances.

In particular, in this paper, we study that given a set of VM pairs and a set of middlebox *software instances* in the data center, how to assign a middlebox instance for each VM pair to traverse, in order to minimize the total energy cost of the VM pairs while satisfying the resource constraint of each middlebox instance. We refer to the problem as load-balanced middlebox assignment problem (LB-MAP). We formulate LB-MAP formally and prove that it is equivalent to the well-known minimum cost flow problem (MCF) in a transformed flow network. MCF can be solved optimally and efficiently [1]. Therefore the LB-MAP can also be solved optimally and efficiently. We also design a suite of efficient heuristic algorithms viz. VM-Based, MB-Based, and VM+MB-Based. Via extensive simulations, we show that all the heuristic algorithms perform close to the optimal minimum cost flow algorithm, while VM+MB-Based performs best among all the heuristic algorithms. To the best of our knowledge, our work is the first that collectively addresses the energy cost minimization for VM communications as well as load-balancing for middleboxes in policy-driven data centers.

II. Related Work

Joseph et al. [19] was one of the first architectural work discussing middlebox management in data centers. It particular, it proposed and designed a policy-aware new layer-2 switching layer in data centers that explicitly forwards different types of traffic through different sequences of middleboxes. They showed that their approach traverses middleboxes correctly as well as efficiently. Qazi et al. [30] further addressed key system design and algorithmic challenges for aforesaid policy enforcement layer using SDN. They proposed efficient data plane support for policy composition, unified switch and middlebox resource management, and automatically dealing with dynamic packet modifications. Sekar et al. [31] took a different perspective and presented a new architecture for middlebox deployments. They proposed to consolidate individual middleboxes as well as their management to multiplex hardware resources and reuse processing modules across different applications.

On the system side, Sherry et al. [32] showed that outsourcing middlebox processing to the cloud relieves enterprises of major problems caused by today's enterprise middlebox infrastructure. Recently Gember et al. [15] realized a software-defined middlebox networking framework by representing, manipulating, and knowledgeably controlling middlebox state. Zhang et al. [35] presented a framework for SDN-enabled services that dynamically route traffic through any sequence of middleboxes. They also proposed an algorithm to select the best locations for placing to optimize the performance.

Middlebox management has strong theoretic roots as well. Liu et al. [23] studied middlebox placement problem, in which given network information and policy specifications, it attempts to determine the optimal locations to place the middleboxes so that either end-to-end delay or the bandwidth consumption is optimized. They showed this problem is NP-hard and proposed two heuristic algorithms. Li et al. [21] studied the policy-aware cloud application embedding problem and designed online primal-dual algorithms. Cui et al. [6], [8], [7], [9] proposed an suite of synergistic schemes to jointly consolidate network policies and virtual machines. They studied dynamic virtual machine consolidation and dynamic network policy (re)allocation to meet both efficient data center resource management and middleboxes traversal requirements.

Among above research, Qazi et al. [30] was the only work that specifically addressed load-balancing issue of middleboxes in data centers. It formulated an online integer linear program (ILP) to minimize the maximum middlebox load across the network. However, as stated in the paper, the load balancing ILP might take a long time for a large network. In contrast, our LB-MAP has different goals and use different solution techniques. We aim to minimize the total energy cost of all the VM pairs in data centers while satisfying the capacities of middleboxes. We propose a time-efficient minimum cost flow optimal solution and a suite of time-efficient heuristic algorithms. Recently, Tu et al. [34] introduced a programmable middlebox that distributes data center traffic more evenly in order to enhance bandwidth utilization and reduce traffic delay. The SDN middlebox controller collects traffic distribution and server loads information and performs load balancing accordingly. However, it did not consider the capacity constraint of each middlebox and did not aim to

minimize the VM communication energy, therefore is different from LB-MAP.

Data Center Topology [3]. We focus on fat-tree networks [3] as they are widely adopted in data centers to interconnect commodity Ethernet switches. Fat tree is a variation of three-stage Clos networks [5], which is rearrangeably non-blocking with 1:1 oversubscription ratio [3]. *Rearrangeably non-blocking* means all the bandwidth available to the end hosts can always be saturated for any communication patterns. *Over-subscription* is the ratio of the worst-case achievable aggregate bandwidth among the end hosts to the total bisection bandwidth of a particular communication topology. An oversubscription of 1:1 indicates that all hosts may potentially communicate with arbitrary other hosts at the full bandwidth of their network interface.

A k-ary fat-tree is shown in Fig. 1 with $k = 4$, where k is the number of ports of each switch. There are three layers of switches: edge switch, aggregation switch and core switch from bottom to top. Core switches handle huge amount of traffic across the entire data center, therefore consuming lots of energy power. In contrast, aggregate switches and edge switches transmit less amount of traffic therefore consuming less power. The lower two layers are separated into k pods. *Pods* are modular units of compute, storage, and networking resources that are designed together as a unit in data center, each containing $k/2$ aggregation switches and $k/2$ edge switches, while forming a complete bipartite graph in between. In particular, each edge switch is directly connected to $k/2$ physical machines (PMs); and each of its remaining $k/2$ ports is connected to each of the $k/2$ aggregation switches from the same pod. There are $\frac{k^2}{4}$ k-port core switches, each of which is connected to each of the k pods. In general, a fat-tree built with k-port switches supports $\frac{k^3}{4}$ PMs. Fig. 1 shows a data center of 16 PMs.

III. Load Balanced Middlebox Assignment Problem (LB-MAP)

A. Problem Formulation.

Network Model. We model a data center as an undirected general graph $G(V, E)$. $V = V_p \cup V_s$ includes the set of physical machines (PMs) V_p and the set of (edge, aggregate, and core) switches V_s. E is the set of edges, each connecting either one switch to another switch or a switch to a PM, as shown in Fig. 1. In the data center network, there are l communicating VM pairs $P = \{(v_1, v_1^{'}), (v_2, v_2^{'}), ..., (v_l, v_l^{'})\}$. v_i and $v_i^{'}$ $(1 \leq i \leq l)$ are referred to as the *source VM* and the *destination VM*, respectively. Each VM v is located in one of the PMs, denoted as $S(v)$. In particular, the PMs $S(v_i)$ and $S(v_i^{'})$ are referred to as the *source PM* and *destination PM* of $(v_i, v_i^{'})$, respectively. A PM can store multiple VMs, and can be source and destination PM simultaneously. In Fig. 1, there are two communicating VM pairs: $(v_1, v_1^{'})$ and $(v_2, v_2^{'})$. Table I shows all the notations used in this paper.

Middlebox Model. According to Gill et al. [16], among all the network devices in data center, load balancers have the highest failure probability. This is due to high number of software faults (such as software bugs and configuration errors) and hardware faults related to application-specific integrated circuit (ASIC) and memory. We assume that there is one type of middlebox in the data center network such as load balancers. However, there are multiple copies of the same middlebox type inside the cloud data centers [12]. We will discuss the more general case wherein there are multiple middlebox types and each has multiple instances such that each VM pair needs to traverse a sequence of middlebox instances in Future Work Section.

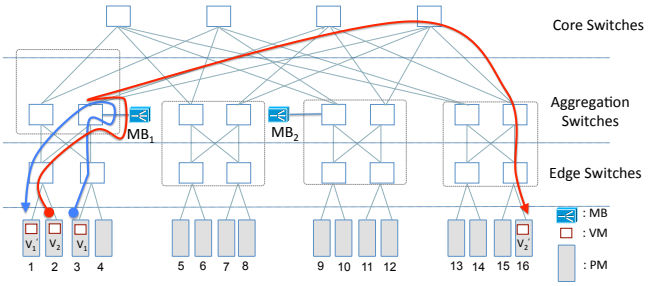

Fig. 1. A k-ary fat tree with $k = 4$ and 16 PMs. There are two communicating VM pairs: $(v_1, v_1^{'})$ and $(v_2, v_2^{'})$, and two middlebox instances MB_1 and MB_2. The capacity of each MB $\kappa = 2$. The minimum VM communication takes place as follows: $(v_1, v_1^{'})$ traverses MB_1 (colored blue, with cost of 3) while $(v_2, v_2^{'})$ traverses MB_1 too (colored red, with cost of 5), resulting in minimum total cost of 8 under uniform energy model.

In particular, we assume there are m software instances of of this middlebox $M = \{mb_1, mb_2, ..., mb_m\}$, which have already been placed inside the data center network. Each instance is placed in one of the switches. Let's assume that mb_j $(1 \leq j \leq m)$ is located at switch $sw(j) \in V_s$. The policy specifies that each communicating VM pair $(v_i, v_i^{'})$ must traverse one of the instances. However, due to the capacity limit of the middlebox, at most κ VM pairs can be served by any middlebox instance. In Fig. 1, there are two load balancer instances: MB_1 and MB_2, the capacity of each MB is $\kappa = 2$.

TABLE I
NOTATION SUMMARY

Notation	Explanation
V_p	The set of physical machines (PMs) in the data center
V_s	The set of switches in the data center
P	The set of l VM communication pairs, $(v_i, v_i^{'})$, $1 \leq i \leq l$
M	The set of m middlebox instances, mb_j, $1 \leq j \leq m$
$S(v)$	The PM where VM v is stored
$sw(j)$	The switch where mb_j is located
κ	The capacity of each middlebox instance
r_e, r_a, r_c	The energy consumption on edge, access, and core switch
$c(i, j)$	The energy cost between PM (or switch) i and j
$c_{i,j}$	The energy cost when VM pair $(v_i, v_i^{'})$ traverses mb_j
C_p	The total energy cost for an MB assignment function p

Bump-Off-The-Wire Design. Traditional middlebox appliances are deployed into the data center network using an inline "bump-in-the-wire" design. In this configuration, dedicated middlebox hardware is plugged in the physical network data path, processing all the traffic passing through it. There are

two drawbacks of this design. First, such on-path deployment of middleboxes forces all traffic on a network path to traverse the same sequence of middleboxes. This is unnecessary since different applications may have different requirements, different application traffic may therefore need to traverse different middleboxes. Second, when there are multiple instances of the same type of middlebox, it is a waste of processing time and hardware resources when the traffic must pass through more than one of them. Instead, we adopt the "bump-off-the-wire" design proposed by Joseph et al. [19], which is further improved by Zhang et al. [35]. In their designs, middleboxes are taken off the physical network paths and implemented as software modules or VMs that are installed on a PM plugged into each switch. Therefore, network traffic is explicitly forwarded to the middleboxes. Due to its low latency links and minimal performance overhead, the data center network is very suitable for such explicit traffic redirection.

Energy Model. We measure the power consumption of any VM pair communication by focusing on the switches it goes through. In particular, we use r_e, r_a, and r_c to denote the power consumption on an edge, aggregate, and core switch respectively, when it transmits one VM communication. We consider two energy consumption models that are currently adopted in cloud data center research.

Uniform Energy Model. In this model, the energy consumption of a VM communication is measured as the minimum number of switches it traverses [25]. That is, it costs the same amount of energy by going through either a core, aggregation, or edge switch: $r_e = r_a = r_c$. For example, in Fig. 1, if $r_e = r_a = r_c = 1$, the power consumption between v_1 and $v_1^{'}$ and between v_2 and $v_2^{'}$ are 3 and 5, respectively.

Skewed Energy Model. This model is based on the fact that the core switches handle more traffic therefore usually consume more energy power than aggregate switches, which consume more energy power than edge switches [4]. Accordingly, we set $r_e < r_a < r_c$. For example, in Fig. 1, if $r_e = 1$, $r_a = 5$, and $r_c = 10$, the power consumption between v_1 and $v_1^{'}$ and between v_2 and $v_2^{'}$ are 7 and 22, respectively.

EXAMPLE 1: In Fig. 1, the capacity of each MB $\kappa = 2$. Each VM pair needs to traverse one of the instances. To minimize the communication cost, $(v_1, v_1^{'})$ traverses MB_1 (colored blue, with cost of 3) while $(v_2, v_2^{'})$ traverses MB_1 too (colored red, with cost of 5), resulting in minimum total communication cost of 8 under uniform energy model. □

Problem Formulation of LB-MAP. Let $c(i, j)$ denote the minimum energy consumption between PM (or switch) i and j. Let $c_{i,j}$ be the minimum power consumption for VM pair $(v_i, v_i^{'})$ when it is assigned to middlebox instance mb_j. Then,

$$c_{i,j} = c\big(S(v_i), sw(j)\big) + c\big(sw(j), S(v_i^{'})\big). \quad (1)$$

Now we define the *load balanced middlebox assignment function* as $p : P \rightarrow M$, signifying that VM pair $(v_i, v_i^{'}) \in P$ is assigned to middlebox instance $p(i) \in M$. Given any middlebox assignment function p, the power consumption for VM pair $(v_i, v_i^{'})$ is then

$$c_{i,p(i)} = c\big(S(v_i), sw(p(i))\big) + c\big(sw(p(i)), S(v_i^{'})\big). \quad (2)$$

Denote the total energy consumption of all the l VM pairs with middlebox assignment p as C^p. Then

$$\begin{aligned} C^p &= \sum_{i=1}^{l} c_{i,p(i)} \\ &= \sum_{i=1}^{l} \Big(c\big(S(v_i), sw(p(i))\big) + c\big(sw(p(i)), S(v_i^{'})\big) \Big). \end{aligned} \quad (3)$$

Let p_{min} be an assignment that yields the minimum total energy consumption among all the middlebox assignments $\mathcal{P}$, i.e., $C^{p_{min}} \leq C^p$, $\forall p \in \mathcal{P}$. The objective of LB-MAP is to find such a p_{min} under the constraint that at most κ VM pairs can be served by any middlebox instance:

$$|\{1 \leq i \leq l | p(i) = j\}| \leq \kappa, \forall j, 1 \leq j \leq m.$$

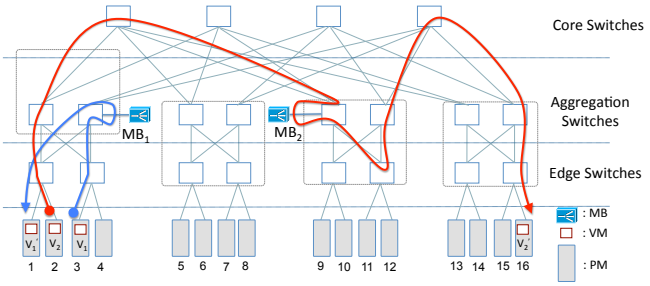

Fig. 2. Load-Balanced VM communication with $\kappa = 1$. The minimum energy communication is then: $(v_1, v_1^{'})$ traverses MB_1 (shown in blue color, with cost of 3) while $(v_2, v_2^{'})$ traversing MB_2 (shown in red color, with cost of 9), resulting in total communication cost of 12 under uniform energy model.

EXAMPLE 2: In Fig. 1, if the capacity of each MB $\kappa = 1$, the two VM pairs can not traverse MB_1 simultaneously. Fig. 2 shows a load-balanced VM pair communication for $\kappa = 1$. $(v_1, v_1^{'})$ traverses MB_1 (shown in blue color, with cost of 3) while $(v_2, v_2^{'})$ traversing MB_2 (shown in red color, with cost of 9), resulting in total communication cost of 12 under uniform energy model. □

B. Minimum Cost Flow (MCF) Optimal Algorithm.

In this subsection, we show that the LB-MAP is equivalent to minimum cost flow problem, which can be solved efficiently and optimally [1]. We first present the minimum cost flow problem with its well-known algorithms, and then transform the data center network to a flow network to show the equivalency.

Minimum Cost Flow Problem (MCF). MCF [1] is a classic optimization problem that finds the cheapest possible way of

sending a certain amount of flow through a flow network, considering that each edge in the flow network has an associated capacity and cost. It is formally defined as below.

Let $G = (V, E)$ be a directed graph. The capacity of edge $(u, v) \in E$, denoted by $c(u, v)$, represents the maximum amount of flow that can pass through this edge. The cost of edge $(u, v) \in E$, denoted by $d(u, v)$, represents the cost when one amount of flow that can pass through an edge. Besides, there is a source node $s \in V$ with b amount of supply, and a sink node $t \in V$ with b amount of demand. A flow on an edge $(u, v) \in E$, denoted by $f(u, v)$, is a mapping $f : E \rightarrow \mathbb{R}^+$, subject to the following two constraints:

1). Capacity constraint: $f(u, v) \leq c(u, v), \forall (u, v) \in E$. That is, the flow of an edge cannot exceed its capacity.

2). Flow conservation constraint: $\sum_{u \in V} f(u, v) = \sum_{u \in V} f(v, u)$, for each $v \in V \setminus \{s, t\}$. That is, the sum of the flows entering a node must equal the sum of the flows exiting a node, except for the source and the sink nodes. The net flow out of source node s is $\sum_{u \in V}(f(s, u) - f(u, s)) = b$; the net flow into sink node t is $\sum_{u \in V}(f(t, u) - f(u, t)) = b$.

The goal of MCF is to find a flow function f such that the total cost of the flow in the network is minimized. That is,

$$\min \Sigma_{(u,v) \in E}\big(d(u, v) \cdot f(u, v)\big). \tag{4}$$

MCF Algorithms. MCF can be solved efficiently by many combinatorial algorithms [1]. There are pseudo-polynomial algorithms including cycle-canceling [27], successive shortest path [10], and out-of-kilter algorithm [14], polynomial algorithms including cost- and capacity scaling [10], [28], [18], and network simplex algorithm [29]. In this paper, we adopt the scaling push-relabel algorithm proposed by Goldberg [17], which works well over a wide range of problem classes. For any flow network, the algorithm has the time complexity of $O(a^2 \cdot b \cdot \log(a \cdot c))$, where a, b, and c are the number of nodes, number of edges, and maximum edge capacity in the flow network, respectively.

Transforming a Data Center Network to a Flow Network. Next, we transform the data center network $G(V, E)$ (Fig. 1) into a new flow network $G'(V', E')$ (Fig. 3). We show that LB-MAP in $G(V, E)$ is equivalent to MCF in $G'(V', E')$. The transformation consists of the following five steps:

Step 1. $V' = \{s_0\} \cup \{t_0\} \cup P \cup M$, where s_0 is the new source node and t_0 is the new sink node in the flow network.

Step 2. $E' = \{(s_0, (v_i, v_i^{'})) : (v_i, v_i^{'}) \in P\} \cup \{((v_i, v_i^{'}), mb_j) : (v_i, v_i^{'}) \in P, mb_j \in M\} \cup \{(mb_j, t_0) : mb_j \in M\}$. Note that it is a complete bipartite graph between P and M.

Step 3. For each edge $(s_0, (v_i, v_i^{'}))$, set its capacity as 1 and its cost 0. For each edge (mb_j, t_0), set its capacity as κ and its cost 0.

Step 4. For each edge $((v_i, v_i^{'}), mb_j)$, $(v_i, v_i^{'}) \in P, mb_j \in M$, set its capacity as 1 and its cost as $c_{i,j}$, the minimum energy cost of VM pair $(v_i, v_i^{'})$ when it is assigned to middlebox instance mb_j.

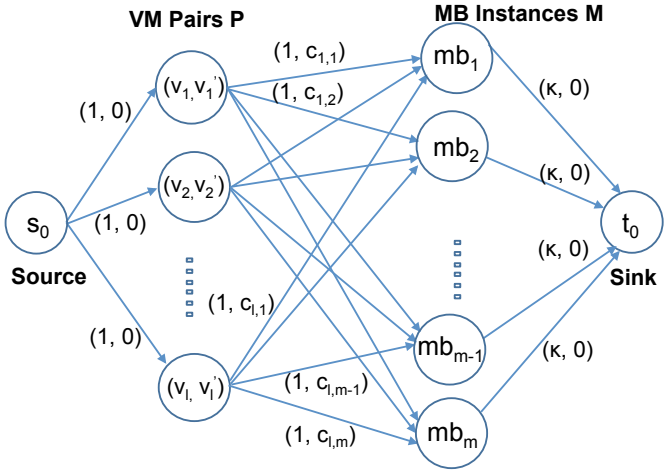

Fig. 3. LB-MAP is equivalent to minimum cost flow problem. In each parenthesis on the edge, the first value is the capacity of the edge and the second the cost of the edge.

Step 5. Set the supply at s_0 and the demand at t_0 as l.

The technique used above is similar to those in [33], [20]. Next, the generated flow network (Fig. 3) is passed to the MCF algorithm discussed above, which outputs the load-balanced middlebox assignment that gives the minimum power consumption for the l VM pairs. That is, for each VM pair, the MCF outputs its assigned middlebox instance without violating the capacity constraint of each middlebox.

Time Complexity of LB-MAP Algorithm. The algorithm consists of two parts: the graph transformation and the MCF algorithm. For the transformation, calculating $c(i, j)$ and $c_{i,j}$ each takes $O(|V|^3)$. Constructing $G'(V', E')$ takes $|V|^3 + l + l \cdot m + m$, which is $O(l \cdot m + |V|^3)$. Therefore it takes $O(l \cdot m + |V|^3)$ for transformation. For the minimum cost flow, the scaling push-relabel algorithm [17] we adopt has the time complexity of $O(a^2 \cdot b \cdot \log(a \cdot c))$, where a, b, and c are the number of nodes, number of edges, and maximum edge capacity in the flow network. From the transformation, $|V'| = l + m + 2$ and $|E'| = l + m + l \cdot m$. Therefore the time complexity of the minimum cost flow is $O\big((l + m)^2 \cdot l \cdot m \cdot \log((l + m) \cdot \kappa)\big)$. The time complexity of the LB-MAP algorithm is thus $O\big(|V|^3 + (l + m)^2 \cdot l \cdot m \cdot \log((l + m) \cdot \kappa)\big)$.

Theorem 1: LB-MAP is equivalent to minimum cost flow problem.

Proof: We show that by applying minimum cost flow algorithm upon the above flow network, it achieves that a) each of the l VM communication pairs is assigned to exactly one middlebox instance while b) satisfying the capacity constraints of middleboxes, and c) achieving the minimum energy consumption for all the l VM pairs.

First, we show that with above transformation, sending l amount of flow from s_0 to t_0 ensures that each of the l VM pairs be assigned to one middlebox instance. In particular, since the amount of supply at s_0 is l (Step 5), and the capacity of each edge $(s_0, (v_i, v_i^{'}))$ $(1 \leq i \leq l)$ is one (Step 3), a

valid flow of l amount from s_0 to t_0 must consist of one amount on edge $(s_0, (v_1, v_1^{'}))$, one amount on $(s_0, (v_2, v_2^{'}))$, ..., and one amount on $(s_0, (v_l, v_l^{'}))$. Now, since the capacity on each edge $((v_i, v_i^{'}), mb_j)$ is one (Step 4), according to flow conservation, then one amount of flow must come out of any edge $(v_i, v_i^{'})$ and go into exactly one of the middlebox instance mb_j. This results in that each VM pair is assigned to exactly one middlebox instance. Note that the capacity of each edge $((v_i, v_i^{'}), mb_j)$ could be set as any positive integers as it does not change above analysis.

Next, we show the VM pair assignment does not violate the capacity constraint of middleboxes. Since the edge capacity of edge (mb_j, t_0) is κ (Step 3), it ensures that no more than κ amount of flow coming out of each node $mb_j \in M$. This guarantees that each middlebox instance mb_j will not accommodate more than κ VM pairs, satisfying the capacity constraint of middleboxes.

Finally, as for the cost, note that the edge cost of $((v_i, v_i^{'}), mb_j)$ is $c_{i,j}$, the minimum energy consumption of VM pair $(v_i, v_i^{'})$ when it is assigned to middlebox mb_j, while all other edges in the flow network have cost zero. This indicates that only the VM communication cost is considered in minimum cost flow. Minimum cost flow algorithm gives the minimum cost of sending l amount of flow from s_0 to t_0, showing that the corresponding VM communication cost obtained is indeed minimum. ∎

C. Three Heuristic Algorithms.

We also propose three polynomial-time greedy algorithms for comparison purpose. Each greedy algorithm takes place in rounds, and is based on a different criterion to choose middlebox for each VM pair.

VM-Based Algorithm. The VM-Based algorithm works as follows. For each VM pair, it is assigned to an MB instance such that it gives the minimum energy consumption for this VM pair among all the MB instances, while satisfying this MB instance's capacity. Finding all the minimum energy consumption paths between all pair of PMs takes $O(|V|^3)$. Assigning each VM pair to an MB instance takes $O(l \cdot m)$. Therefore the time complexity of VM-Based Algorithm is $O(|V|^3 + l \cdot m)$.

Algorithm 1: VM-Based Algorithm.
Input: A data center $G(V, E)$ with l VM pairs and m MBs
Output: Total energy cost C for all the l VM pairs.
Notations:
i: the index for VM pairs
j: the index for middlebox instances
$load(j) = 0$: the current load of mb_j
c^i_{min}: the minimum energy cost for VM pair $(v_i, v_i^{'})$
j^*: middlebox mb_{j^*} is assigned to $(v_i, v_i^{'})$

1. $C = 0$;
2. **for** ($i = 1$ to l)
3. $c^i_{min} = infinite$;
4. **for** ($j = 1$ to m)
5. **if** ($c(i,j) \leq c^i_{min}$ and $load(j) < \kappa$)
6. $c^i_{min} = c(i,j)$;
7. $j^* = j$;
8. **end if;**
9. **end for;**
10. $load(j^*)$++;
11. $C = C + c^i_{min}$;
12. **end for;**
13. **RETURN** C.

MB-Based Algorithm. In this algorithm, for each MB instance, it is assigned κ VM pairs among all the VM pairs that give the minimum energy consumption when going through that MB instance. The running time of this algorithm is $|V|^3 + m \cdot (l + l \cdot \lg l + \kappa)$, which is $O(|V|^3 + m \cdot l \cdot \lg l)$.

Algorithm 2: MB-Based Algorithm.
Input: A data center $G(V, E)$ with l VM pairs and m MBs
Output: Total energy cost C for all the l VM pairs.
Notations:
i: the index for VM pairs
j: the index for middlebox instances
X_j: the set of VM pairs assigned to mb_j
$assigned[i]$: true if $(v_i, v_i^{'})$ is assigned, false if not

1. $C = 0$;
2. **for** ($i = 1$ to l)
3. $assigned[i] = false$;
4. **end for;**
5. **for** ($j = 1$ to m)
6. $X_j = \phi$;
7. **for** ($i = 1$ to l)
8. **if** ($assigned[i] == false$)
9. $X_j = \{(i, c(i,j))\} \cup X_j$;
10. **end if;**
11. **end for;**
12. Sort X_j in the non-descending order of $c(i,j)$;
13. $X_j = \{(x_1, c(x_1,j)), (x_2, c(x_2,j)), (x_3, c(x_3,j)), ...\}$,
14. where $c(x_1,j) \leq c(x_2,j) \leq c(x_3,j)...$;
15. **for** ($k = 1$ to κ)
16. $C = C + c(x_k, j)$;
17. **end for;**
18. **end for;**
19. **RETURN** C.

VM-MB-Based Algorithm. In each round, it checks which VM pair is assigned to which MB instance, such that when that VM pair traverses that MB instance, it yields the minimum energy consumption among all the unassigned VM pairs and all the MB instances in that round. The time complexity of this algorithm is $O(|V|^3 + m \cdot l)$.

Algorithm 3: VM-MB-Based Algorithm.
Input: A data center $G(V, E)$ with l VM pairs and m MBs
Output: Total energy cost C for all the l VM pairs.
Notations:
i: the index for VM pairs
j: the index for middlebox instances
$load(j) = 0$: the current load of mb_j

j^*: middlebox mb_{j^*} is assigned in each round,
c_{min}: the minimum energy obtained in each round
1. $C = 0$;
2. **for** ($i = 1$ to l)
3. $assigned[i] = false$;
4. **end for;**
5. **while** (there are still unassigned VM pairs)
6. $c_{min} = infinite$;
7. **for** ($i = 1$ to l)
8. **if** ($assigned[i] == false$)
9. **for** ($j = 1$ to m)
10. **if** ($load(j) < \kappa$ and $c(i,j) \leq c_{min}$)
11. $c_{min} = c(i,j)$;
12. $j^* = j$;
13. **end if;**
14. **end for;**
15. **end if;**
16. **end for;**
17. $load(j^*)$++;
18. $C = C + c_{min}$;
19. **end while;**
20. **RETURN** C.

IV. Performance Evaluation

Simulation Setting. In this section, we compare the performances of the four LB-MAP algorithms: Minimum Cost Flow Algorithm (referred to as **MCF**), VM-Based Algorithm (referred to as **VM**), MB-Based (referred to as **MB**), and VM+MB-Based Algorithm (referred to as **VM+MB**). We create data centers of two different sizes: $k = 8$, a small data center with 128 PMs and $k = 16$, a relatively large-size data center with 1024 PMs. The source and destination VMs of each VM pair are randomly placed on the PMs and the MB instances are randomly placed on the switches. In all the simulation plots, each data point is an average of 10 runs, and the error bars indicate 95% of confidence interval. For fair comparison, for each simulation run, we run the four algorithms on the same network set up with the same initial placement of VM communication pairs and MB instances.

In all the simulations, unless otherwise mentioned, we vary the number of VM pairs in the data center l from 100, 200, ..., to 500, and number of MB instances m from 1 to 2 to 3. In each case, the middlebox capacity $\kappa = \lceil \frac{l}{m} \rceil$. Throughout the simulation, we set $r_e = r_a = r_c = 1$ for the uniform energy model and $r_e = 1$, $r_a = 5$, and $r_c = 10$ for the skewed energy model. Recall that r_e, r_a, and r_c denote the power consumption on the edge, aggregate, and core switches respectively.

Effect of Number of VM Pairs l. We first investigate the effect of the number of VM pairs on the total energy cost, shown in Fig. 4. We set the number of MB instances in the data center as 3 and vary the number of VM pairs from 100, 200, ..., to 500. We have several observations. First, for each algorithm, with the increase of number of VM pairs, the total energy cost increases as well. This is obvious since more VM pairs in general incurs more energy consumption since all the VM pairs are randomly placed. Second, the performance order of the four algorithms are (from worst to best) is VM-Based, MB-Based, VM+MB-Based, and MCF. This indirectly supports our theoretical proof that the minimum cost flow algorithm is optimal, performing better that the other three heuristic algorithms. With the increase of number of VM pairs, the performance difference between MCF and the other three seems to be more evident.

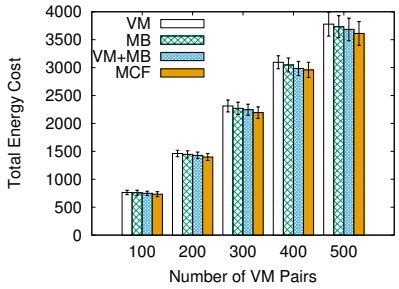

Fig. 4. Varying number of VM pairs under uniform energy model. Here, number of MBs $m = 3$, number of PMs in data center is 128.

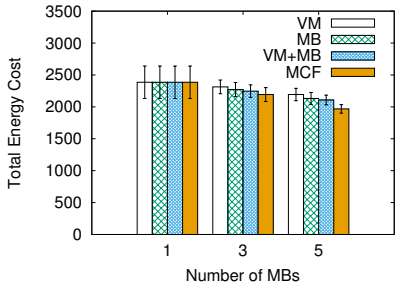

Fig. 5. Varying number of MB instances under uniform energy model. Here, number of VM pairs $l = 300$, number of PMs in data center is 128.

Effect of Number of MB Instances m. We then investigate the effect of the number of MB instances on the total energy cost, shown in Fig. 5. We set the number of VM pairs in the data center as 300 and vary the number of MB instances as 1, 2, 3. We have several observations. First, for each algorithm, with the increase of number of MB instances, the total energy cost decreases. This is because more MB instances give each VM pairs more options to choose an energy-efficient path between source and destination VMs, incurring less energy consumption. Second, when there is only one middlebox, it is observed that all the four algorithms perform the same. In this case, all the VM pairs must go through the same middlebox for policy requirement independent of the algorithms, therefore costing the same amount of energy for all four algorithms. Finally, we still observe that MCF performs the best, yielding the minimum amount of energy consumption. Among the three heuristic algorithms, VM+MB-based performs better than MB-based, which performs better than VM-based.

Comparison in Large Data Centers. Fig. 6 and Fig. 7 show

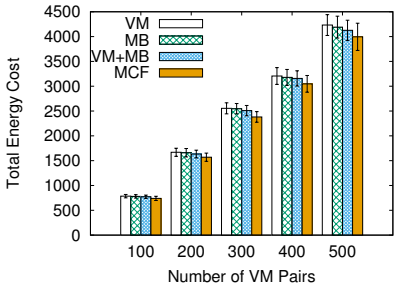

Fig. 6. Varying number of VM pairs under uniform energy model. Here, number of MBs $m = 3$, number of PMs in data center is 1024.

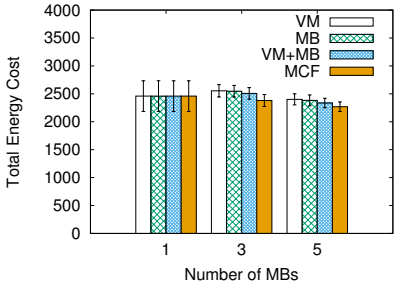

Fig. 7. Varying number of MB instances under uniform energy model. Here, number of VM pairs $l = 300$, number of PMs in data center is 1024.

the performance comparison in a large data center of 1024 PMs, by varying number of VM pairs and number of middleboxes, respectively. We have the same general observations as in a smaller data enter shown in Fig. 4 and Fig. 5, however, with one notable difference. Fig. 7 shows that for all the three heuristic algorithms, when number of MB instances increases from one to three, their energy costs actually increase. This is counter-intuitive at first, since more middleboxes help with reducing the energy cost. However, this increase is due to the random initial placement of the VM pairs and middleboxes, under which it is possible that a placement with two MBs does not help as much as a placement of one MB for different set of randomly placed VM pairs.

Comparison Under Skewed Energy Model. Finally we investigate how the skewed energy model affects the energy costs of the four algorithms. Fig. 8 and Fig. 9 show the performance comparison in a small data center of 128 PMs, while Fig. 10 and Fig. 11 show the performance comparison in a large data center of 1024 PMs. We observe again that MCF performs best, yielding the minimum amount of energy in all three algorithms. VM+MB-Based performs better than the other two heuristic algorithms. This is mainly because in each round, it selects an assigned VM pair and an MB so that when this pair is assigned to this MB, it yields the smallest energy cost among all the unassigned VM pairs and MB combination. Whereas for the other two algorithms (VM-Based and MB-Based), it only focuses on one of them without checking thoroughly all the possibilities.

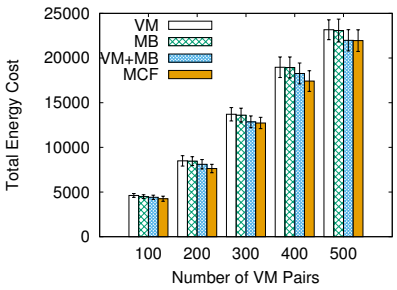

Fig. 8. Varying number of VM pairs under skewed energy model. Here, number of MBs $m = 3$, number of PMs in data center is 128.

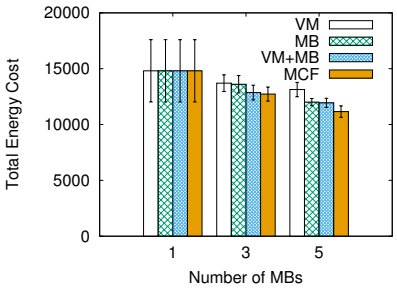

Fig. 9. Varying number of MB instances under skewed energy model. Here, number of VM pairs $l = 300$, number of PMs in data center is 128.

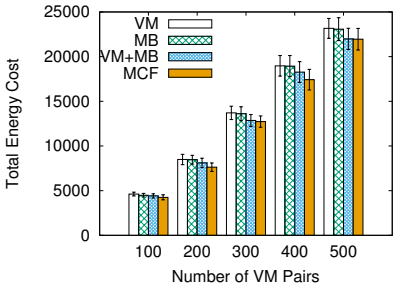

Fig. 10. Varying number of VM pairs under skewed energy model. Here, number of MBs $m = 3$, number of PMs in data center is 1024.

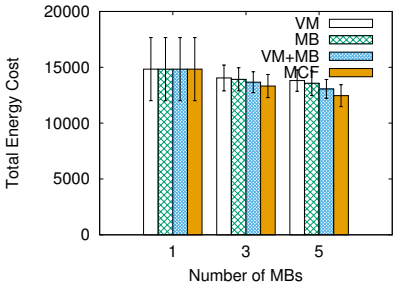

Fig. 11. Varying number of MB instances under skewed energy model. Here, number of VM pairs $l = 300$, number of PMs in data center is 1024.

V. Conclusion and Future Work

We studied LB-MAP: Load-Balanced Middlebox Assignment Problem in policy-driven data centers. The goal of LB-MAP is to minimize the energy cost of all the communicating virtual machine pairs who must traverse a middlebox for policy requirement, while taking into account of the limited capacity of the middlebox. Much research has been done to study how to place middleboxes inside data centers for cost-efficient VM communication. However, not much research has focused on load balance of middleboxes. We formulated LB-MAP formally and proved that LB-MAP is equivalent to the well-known minimum cost flow problem (MCF), which can be solved optimally and efficiently. We also designed a suite of efficient heuristic algorithms based on different criteria viz. VM-Based, MB-Based, and VM+MB-Based. Via extensive simulations, we showed that all the heuristic algorithms perform close to the optimal minimum cost flow algorithm, while VM+MB-Based performs best among all the heuristic algorithms. To the best of our knowledge, this is the first work that addresses the energy cost minimization for VM communications as well as load-balancing for middleboxes in policy-driven data centers.

In our current setup, we assume that there is only one middlebox type such as load balancers. In the future, we will consider a more general problem wherein multiple types of middleboxes (e.x., load balancers, firewalls, and IDSs) exist, each having multiple instances. To satisfy policy requirement, each VM pair needs to traverse a sequence of middlebox instances of different types in a particular order. In this so called *service chain* scenario, tackling middlebox load-balancing as well as minimizing VM pair communication cost become more challenging problems.

References

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., 1993.

[2] Ian F. Akyildiz, A. Lee, P. Wang, M. Luo, and W. Chou. A roadmap for traffic engineering in sdn-openflow networks. *Computer Networks (Elsevier)*, 71:1–30, 2014.

[3] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. *SIGCOMM Comput. Commun. Rev.*, 38(4):63–74, August 2008.

[4] D. Boru, D. Kliazovich, F. Granelli, P. Bouvry, and A. Y. Zomaya. Energy-efficient data replication in cloud computing datacenters. *Cluster Computing*, 18(1):385–402, 2015.

[5] C. Clos. A study of non-blocking switching networks. *Bell System Technical Journal*, 32(2), 1953.

[6] L. Cui, R. Cziva, F. P. Tso, and D. P. Pezaros. Synergistic policy and virtual machine consolidation in cloud data centers. In *Proc. of IEEE INFOCOM*, 2016.

[7] L. Cui and F. P. Tso. Joint virtual machine and network policy consolidation in cloud data centers. In *Proc. of IEEE CloudNet*, 2015.

[8] L. Cui, F. P. Tso, D. P. Pezaros, and W. Jia. Plan: A policy-aware vm management scheme for cloud data centres. In *Proc. of IEEE/ACM UCC*, 2015.

[9] L. Cui, F. P. Tso, D. P. Pezaros, W. Jia, and W. Zhao. Policy-aware virtual machine management in data center networks. In *Proc. of IEEE ICDCS*, 2015.

[10] J. Edmonds and R.M. Karp. Theoretical improvements in algorithmic efficiency for network flow problems. *Journal of ACM*, 19:248–264, 1972.

[11] R. Guerzoni et al. Network functions virtualisation: an introduction, benefits, enablers, challenges and call for action, introductory white paper. In *SDN and OpenFlow World Congress*, 2012.

[12] A. Jukan F. Carpio, S. Dhahri. Vnf placement with replication for load balancing in nfv networks. In *IEEE ICC*, 2017.

[13] N. Feamster, J. Rexford, and E. Zegura. The road to sdn. *Queue*, 11(12):20–40, 2013.

[14] D.R. Fukerson. An out-of-kilter method for minimal-cost flow problems. *Journal of the Society for Industrial and Applied Mathematics*, 9(1):18–27, 1961.

[15] A. Gember, P. Prabhu, Z. Ghadiyali, and A. Akella. Toward software-defined middlebox networking. In *Proc. of HotNets-XI*, 2012.

[16] P. Gill, N. Jain, and N. Nagappan. Understanding network failures in data centers: Measurement, analysis, and implications. *SIGCOMM Comput. Commun. Rev.*, 41(4):350–361, 2011.

[17] A. V. Goldberg. An efficient implementation of a scaling minimum-cost flow algorithm. *J. Algorithms*, 22:1–29, 1997.

[18] A. V. Goldberg and R. E. Tarjan. Finding minimum-cost circulations by successive approximation. *Math. Oper. Res.*, 15(3):430–466, 1990.

[19] D. A. Joseph, A. Tavakoli, and I. Stoica. A policy-aware switching layer for data centers. In *Proc. of the ACM SIGCOMM*, 2008.

[20] P. Khani, B. Tang, J. Han, and M. Beheshti. Dao-r: Integrating data aggregation and offloading in sensor networks via data replication. In *Proceedings of IEEE GLOBECOM 2015*.

[21] L. E. Li, V. Liaghat, H. Zhao, M. Hajiaghayi, D. Li, G. Wilfong, Y. R. Yang, and C. Guo. Pace: Policy-aware application cloud embedding. In *Proc. of IEEE INFOCOM*, 2013.

[22] Y. Li and M. Chen. Software-defined network function virtualization: A survey. *IEEE Access*, 3, 2015.

[23] J. Liu, Y. Li, Y. Zhang, L. Su, and D. Jin. Improve service chaining performance with optimized middlebox placement. *IEEE Transactions on Services Computing*, 2015.

[24] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, 38(2):69–74, 2008.

[25] X. Meng, V. Pappas, and L. Zhang. Improving the scalability of data center networks with traffic-aware virtual machine placement. In *Proc. of IEEE INFOCOM*, 2010.

[26] R. Mijumbi, J. Serrat, J. L. Gorricho, N. Bouten, F. D. Turck, and R. Boutaba. Network function virtualization: State-of-the-art and research challenges. *IEEE Communications Surveys and Tutorials*, 18(1), 2015.

[27] K. Morton. A primal method for minimal cost flows with applications to the assignment and transportation problems. *Management Science*, 14:205–220, 1967.

[28] J. B. Orlin. A faster strongly polynominal minimum cost flow algorithm. In *In Proc. of the 20th ACM Symposium on the Theory of Computing*, pages 377–387, 1988.

[29] J. B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. *Mathematical Programming*, 79:109–129, 1997.

[30] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu. Simplefying middlebox policy enforcement using sdn. In *Proc. of the ACM SIGCOMM*, 2013.

[31] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi. Design and implementation of a consolidated middlebox architecture. In *Proc. of NSDI*, 2012.

[32] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar. Making middleboxes someone else's problem: Network processing as a cloud service. In *Proc.eedings of the ACM SIGCOMM*, 2012.

[33] B. Tang, N. Jaggi, and M. Takahashi. Achieving data k-availability in intermittently connected sensor networks. In *Proceedings of the International Conference on Computer Communications and Networks (ICCCN'14)*.

[34] R. Tu, X. Wang, J. Zhao, Y. Yang, L. Shi, and T. Wolf. Design of a load-balancing middlebox based on sdn for data centers. In *IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2015.

[35] Y. Zhang, N. Beheshti, L. Beliveau, G. Lefebvre, R. Manghirmalani, R. Mishra, R. Patney, M. Shirazipour, R. Subrahmaniam, C. Truchan, and M. Tatipamula. Steering: A software-defined networking for inline service chaining. In *Proc. of the IEEE ICNP*, 2013.