FMDV: Dynamic Flow Migration in Virtual Network Function-Enabled Cloud Data Centers

Phillip Aguilera, Christopher Gonzalez, Bin Tang
Department of Computer Science
California State University Dominguez Hills, Carson, CA 90747, USA
Email: {paguilera5,cgonzalez393}toromail.csudh.edu, btang@csudh.edu

Abstract—**Virtual Network Functions (VNFs) are software implementation of middleboxes (MBs) (e.g., firewalls and proxy servers) that provide performance and security guarantees for virtual machine (VM) cloud applications. In this paper, we study a new VM flow migration problem for dynamic VNF-enabled cloud data centers (VDCs). The goal is to migrate the VM flows in the dynamic VDCs to minimize the total network traffic while load-balancing VNFs with limited processing capabilities. We refer to the problem as FMDV: flow migration in dynamic VDCs. We propose an optimal and efficient minimum cost flow-based flow migration algorithm and two benefit-based efficient heuristic algorithms to solve the FMDV. Via extensive simulations, we show that our algorithms are effective in mitigating dynamic cloud traffic while achieving load balance among VNFs. In particular, all our algorithms reduce dynamic network traffic in all cases and our optimal algorithm always achieves the best traffic-mitigation effect, reducing the network traffic by up to 28% compared to the case without flow migration.**

Keywords – Virtual Network Functions, Load-Balancing, Flow Migration, Dynamic Cloud Data Centers

I. Introduction

Background. One of the building blocks of modern cloud data centers is virtualization, wherein cloud user applications previously running on multiple physical machines (PMs) can now run as virtual machines (VMs) in a single PM [7]. The benefits of virtualization include reduced IT expenses, enhanced resiliency of IT infrastructure, and increased efficiency and productivity. Recently, Network Function Virtualization (NFV) has become an effective virtualization technique that achieves flexible cloud service management in the cloud computing environment [15]. With NFV, proprietary hardware middleboxes (MBs) such as firewalls and cache proxies can now be implemented as virtual network functions (VNFs) running as lightweight containers on commodity hardware [14]. Being provisioned as different services in cloud data centers, VNFs provide security and performance guarantees to cloud user applications in a flexible and cost-effective manner. We refer to the cloud data centers that implement VNFs to provide network services as *VNF-enabled data centers (VDCs)*.

While the hardware MBs have dedicated hardware resources such as CPU, memory, or accelerators, software VNFs usually have less packet processing capability and are more prone to software bugs, malfunctions, and misconfiguration. As such, VNFs can be more easily overloaded by the high VM cloud traffic and cause packet loss and traffic delay in VDCs. Therefore, how to load-balance the VNFs is an important problem in VDCs.

Fortunately, the software implementation of VNFs makes it possible to replicate and place multiple VNF instances easily inside the VDCs [9]. With multiple instances of the same VNF, the VM cloud traffic in VDCs just needs to visit one of the instances to achieve the security and performance guarantees brought by the VNF. By distributing VM network traffic among multiple VNF instances, it not only achieves load-balance to cloud network traffic thus reducing cloud network congestions but also achieves load-balance to VNF instances thus prolonging their functional lifetime.

Dynamic and Diverse Cloud Traffic. Recent reports about Facebook and other production data centers observe that VM traffic loads (i.e., transmission rates and bandwidth demands) are highly diverse and dynamic among different cloud user applications. It shows that the various and distinct services in data centers exhibit different traffic patterns and the heavy hitters are bound to bursty and rapidly changing [18]. Another recent example is Zoom cloud conferencing [1], where one Zoom Meeting Connector VM [2] could support conference meetings from a few participants of low traffic rates to up to 1000 participants of high traffic rates. Different Zoom meetings could last from minutes to hours in the form of videos, voices, or chat texts and consume dramatically different amounts of network bandwidth. Such dynamic and diverse traffic, if not dealt with well, could deteriorate the utilization of cloud resources such as bandwidth and energy in VDCs.

Our Contributions. We propose migrating VM flows to alleviate the dynamic traffic in cloud data centers. In particular, we identify, formulate and solve a new VM flow migration problem called FMDV: flow migration in dynamic VDCs. The goal of FMDV is to migrate the VM flows in the dynamic VDCs to minimize the total network traffic while load-balancing VNFs with limited processing capability. We propose an optimal minimum cost flow-based algorithm and two benefit-based efficient heuristics to solve the FMDV. Via extensive simulations, we show that our algorithms are effective in optimizing the network resources as well as achieving load balance among VNFs. In particular, all our algorithms can reduce dynamic network traffic in all cases and our optimal algorithm always achieves the best traffic-mitigation effect, reducing the network traffic by 28% compared to the case without flow migration.

II. Related Work

Migrating active in-process flows among VNFs has been active research in recent years. Some research focused on guaranteeing loss-free and order-preserving flow states and packets. For example, Wang et al. [22] designed a distributed flow migration framework to decouple the state transfer and packets migrations, which allows optimizing the two processes separately and in parallel. Another issue is VNF elasticity control, which studies how to scale out, scale in, and load-balance VNFs depending on the traffic load. Sun et al. [19] built a flow migration controller to achieve VNF elasticity control by selecting flows for migrations with the goal of minimizing the load variance of VNF instances. Qazi et al. [16] proposed to minimize the maximum load of a VNF to achieve VNF load-balancing and showed it to be NP-hard.

In contrast, our flow migration techniques are specially designed to mitigate dynamic network scenarios wherein the traffic rates of VM flows are changing. We achieve load-balancing of VNFs by specifying that each VNF has a limited processing capacity. Therefore, instead of minimizing the maximum load or the load variance of VNF instances, our goal is to migrate the flows among VNFs to minimize their total communication cost while respecting the capacity constraint of VNFs. We are able to propose a time-efficient minimum cost flow-based optimal solution.

Qu et al. [17] presented a dynamic flow migration problem for SDN/NFV-enabled 5G communication systems. It considered service function chains (SFCs) and formulated a multi-objective mixed-integer optimization problem that addresses the trade-off between load-balancing and reconfiguration overhead of SFCs. As it is NP-hard, it presented an effective heuristic algorithm that does not have a performance guarantee. In contrast, by assuming each VM flow only accesses one VNF instance, we are able to come up with an optimal and efficient dynamic flow migration scheme that optimizes cloud resources in a dynamic environment.

Alqarni et al. [6] proposed an MB assignment problem in policy-driven data centers. The goal is to assign VM flows to MBs to minimize the communication cost of all the VM flows. However, it assumed that all VM flows have the same traffic rates. More importantly, it did not consider the dynamic cloud traffic addressed in this paper. We instead propose migrating VM flows to mitigate the dynamic cloud traffic. Some preliminary results of this work appeared in an undergraduate research conference [3], in which only two benefit-based flow migration techniques were proposed. However, both algorithms are heuristic algorithms that do not provide any performance guarantee. This paper proposes a minimum cost flow optimal and efficient flow migration scheme that always outperforms the two heuristics.

There are works that employed machine learning techniques to estimate network traffic rates in order to adjust VNF deployment [10], [23], [21]. Another line of work is by Cui et al. [8] and Flores et al. [11], which proposed migrating VMs instead of VNFs to ameliorate dynamic traffic in cloud data centers. Our work instead focuses on migrating the VM flows while addressing the load-balancing issue of VNFs, which is different from all the existing work.

III. Problem Formulation of FMDV

Network Model. We model a VDC as an undirected general graph $G(V, E)$. $V = V_p \cup V_s$ includes a set V_p of PMs and a set V_s of switches, and E is the set of edges. We use fat-trees [5] to illustrate the problem and its solutions, but our problem and solutions are applicable to any data center topology. There are l communicating VM flows $Q = \{q_1, q_2, ..., q_l\}$, where flow $q_i = (v_i, v_i')$ consists of two VMs viz. v_i and v_i' that communicate with each other following some traffic rates. The traffic rate of a flow is the communication frequency or bandwidth demand of this flow. Let $\mathcal{V} = \{v_1, v_1', v_2, v_2', ..., v_l, v_l'\}$ and $v \in \mathcal{V}$ is located $s(v) \in V_p$.

Let λ_i denote the traffic rate of q_i at some moment and $\overrightarrow{\lambda} = \langle \lambda_1, \lambda_2, ..., \lambda_l \rangle$ the *traffic rate vector* of the l VM flows. In a dynamic VDC, the traffic rate of a VM flow changes over time, $\overrightarrow{\lambda}$ is thus not a constant. Fig. 1 shows a fat-tree with 16 PMs and two VM flows: $q_1 = (v_1, v_1')$ and $q_2 = (v_2, v_2')$ with initial traffic rates $\overrightarrow{\lambda} = \langle 100, 1 \rangle$.

VNF Model. There are m VNF instances $M = \{vnf_1, vnf_2, ..., vnf_m\}$ in the VDC. For security and performance purposes, each communicating VM flow q_i must visit one of the VNF instances. We assume that each switch is attached with a server that can install VNFs [24]. We also assume that the VNFs are installed on servers of different switches; that is, vnf_j is installed on switch $w(j) \in V_s$ and $w(j) \neq w(j')$ if $j \neq j'$. The processing capacity of vnf_j is κ_j, meaning it can process at most κ_j VM flows at the same time. Obviously, we have $\sum_{j=1}^{m} \kappa_j \geq l$. Fig. 1 shows a fat tree-based VDC graph with two VNF instances: vnf_1 and vnf_2 with $\kappa_1 = \kappa_2 = 1$. Table I shows all the notations.

TABLE I
NOTATION SUMMARY

Notation	Explanation
V_p	The set of PMs in a VDC
V_s	The set of switches in a VDC
Q	The set of l VM flows in a VDC, $q_i = (v_i, v_i')$
λ_i	Traffic rate of q_i, $1 \leq i \leq l$
M	The set of m VNF instances, vnf_j, $1 \leq j \leq m$
$s(v)$	The PM where VM v is located
$w(j)$	The switch where vnf_j is installed
κ_j	The processing capacity of vnf_j
$w(u, v)$	The weight of edge $(u, v) \in E$
$c(u, v)$	The cost between any two nodes u and v in a VDC
$c_{i,j}$	The communication cost of q_i when traversing vnf_j
$\tau(i, f(i))$	The total migration and communication cost of q_i under flow migration scheme f
C_c^p	The total communication cost of all VM pairs at initial VNF assignment p
C_c^f	The total communication cost after flow migration f
C_m^f	The total flow migration cost under migration f
C_t^f	The total comm. and migra. cost; $C_t^f = C_m^f + C_c^f$

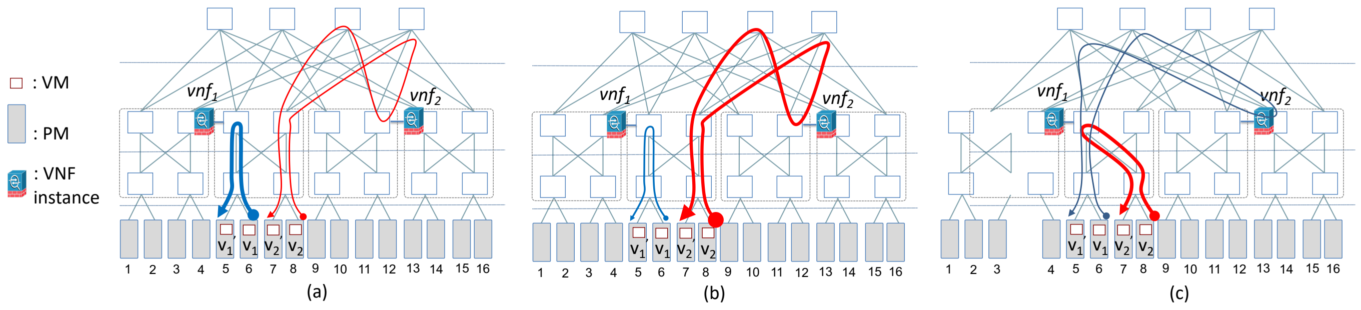

Fig. 1. A VDC graph $G(V, E)$, which is a k-ary fat tree with $k = 4$ and 16 PMs. There are two communicating VM flows: (v_1, v_1') and (v_2, v_2') with initial traffic rate vector $\langle 100, 1\rangle$, and two VNF instances: vnf_1 and vnf_2. The capacities of VNFs $\kappa_1 = \kappa_2 = 1$.

Cost Model. Each edge $(u, v) \in E$ has a cost $w_{u,v}$, indicating either the delay or energy cost on this edge for one unit of VM communication or flow migration. Given any PM or switch u and v, let $c(u, v)$ denote the cost of the shortest path between u to v. Let $c_{i,j}$ be the *communication cost* for VM flow q_i when it visits vnf_j; $c_{i,j} = \lambda_i \cdot \Big(c\big(s(v_i), w(j)\big) + c\big(w(j), s(v_i')\big)\Big)$. The *flow migration cost* of migrating any VM flow from vnf_i to vnf_j is $\mu \cdot c\big(w(i), w(j)\big)$. Here μ is *flow migration coefficient*, which is the ratio between costs of VM flow migration and VM communication. It represents the relative size of memory or data packet transferred in VM flow migration and VM communication.

Let $p : [1, 2, ..., l] \to [1, 2, ..., m]$ denote the *initial VNF assignment*, indicating that $q_i \in Q$ is currently visiting $vnf_{p(i)} \in M$ while the capacity constraints of VNFs are satisfied: $|\{1 \le i \le l | p(i) = j\}| \le \kappa_j, 1 \le j \le m$. The communication cost of q_i with p is then $c_{i,p(i)} = \lambda_i \cdot \Big(c\big(s(v_i), w(p(i))\big) + c\big(w(p(i)), s(v_i')\big)\Big)$. Denote the total communication cost of all the l VM flows with p as C_c^p. $C_c^p = \sum_{i=1}^{l} \lambda_i \cdot \Big(c\big(s(v_i), w(p(i))\big) + c\big(w(p(i)), s(v_i')\big)\Big)$.

EXAMPLE 1: In Fig. 1(a), with initial traffic rate vector of $\langle 100, 1\rangle$, the optimal VNF assignment is that (v_1, v_1') traverses vnf_1 following dark blue line while (v_2, v_2') traverses vnf_2 following light red line, resulting in minimum total cost of $100 \times 4 + 1 \times 8 = 408$. Next, in Fig. 1(b), due to dynamic traffic, the traffic rate vector changes to $\langle 1, 100\rangle$. The resultant VM communication cost becomes $1 \times 4 + 100 \times 8 = 804$, a dramatic and an almost 100% increase. □

Therefore, there is a need to migrate the VM flows from one VNF instance to another in order to reduce the network traffic while still satisfying the capacity constraints of VNFs. As migrating flows incurs network traffic and cost, we need to find an optimal flow migration scheme to minimize the total VM flow migration and communication cost, as defined below.

Problem Formulation of FMDV. We define a *flow migration function* as $f : [1, 2., ...l] \to [1, 2, ..., m]$, meaning that the flow q_i will be migrated from its current VNF $vnf_{p(i)}$ to another VNF $vnf_{f(i)}$. $f(i) = p(i)$ means the flow of (v_i, v_i') does not migrate. Let $C_m^f = \mu \cdot \sum_{i=1}^{l} c\big(p(i), f(i)\big)$ be the *total migration cost* of all the l VM flows with flow migration f. Let C_c^f be the *total communication cost* of all VM flows *after* VM flow migration f. Let C_t^f be the *total cost* of VM flow migration and communication after flow migration f. Then,

$$\begin{aligned} C_t^f &= C_m^f + C_c^f \\ &= \mu \cdot \sum_{i=1}^{l} c\big(p(i), f(i)\big) + \\ &\sum_{i=1}^{l} \Big(c\big(s(v_i), w(p(i))\big) + c\big(w(p(i)), s(v_i')\big)\Big). \end{aligned} \quad (1)$$

Let $\tau(i, f(i)) = \mu \cdot c\big(p(i), f(i)\big) + \Big(c\big(s(v_i), w(p(i))\big) + c\big(w(p(i)), s(v_i')\big)\Big)$, which is the total migration and communication cost of flow q_i with migration scheme f. Then $C_t^f = \sum_{i=1}^{l} \tau(i, f(i))$. The objective of FMDV is to find a flow migration f such that C_t^f is minimized under the processing capacity constraint of VNFs: $|\{1 \le i \le l | f(i) = j\}| \le \kappa_j, 1 \le j \le m$.

EXAMPLE 2: Continuing with Example 1, Fig. 1(c) shows that if we migrate (v_1, v_1') to visit vnf_2 and (v_2, v_2') to visit vnf_1, the total communication cost reduces to $1 \times 8 + 100 \times 4 = 408$. Assuming $\mu = 10$, the incurred flow migration cost is $10 \times 2 + 10 \times 2 = 40$. So total cost is $408 + 40 = 448$, a 44.3% decrease compared to 804 before flow migration. □

IV. Algorithmic Solutions of FMDV

In this section, we present one optimal and efficient algorithm and two efficient heuristic algorithms for FMDV.

A. *Optimal Algorithm.*

First, we show that FMDV in a VDC graph is equivalent to a minimum cost flow problem (MCF) [4] in a properly converted flow network. Given a directed graph $G' = (V', E')$ with a source node s and a sink node t, each edge $(u, v) \in E'$ has a capacity $c(u, v)$ as well as a cost $d(u, v)$, and $f(u, v)$ is the flow on an edge $(u, v) \in E'$. The goal of MCF is to find a flow function f to minimize the total cost of transmitting y amount of flow from s to t, i.e. $\Sigma_{(u,v) \in E'} \big(d(u, v) \cdot f(u, v)\big)$, subject to (a) capacity constraint: $f(u, v) \le c(u, v), \forall (u, v) \in E'$, (b) flow conservation constraint: $\sum_{u \in V} f(u, v) = \sum_{u \in V} f(v, u)$, for each $v \in V - \{s, t\}$, and (c) the net flow out of s and the net flow into t are both y. MCF can be solved efficiently by many combinatorial algorithms [4]. We adopt the scaling

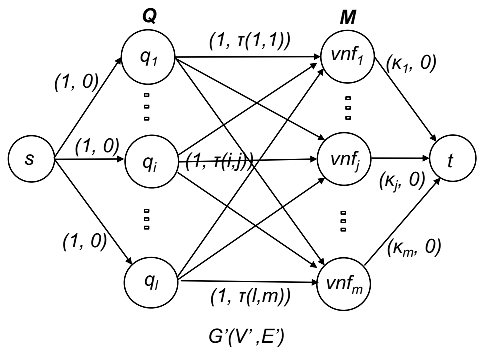

Fig. 2. The flow network $G'(V', E')$ converted from $G(V, E)$ in Fig. 1.

push-relabel algorithm proposed by Goldberg [12], as it has the highest performance codes available for such network optimization. It 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 of $G'(V', E')$.

Fig. 2 shows how to convert the VDC graph $G(V, E)$ in Fig. 1 into a new flow network $G'(V', E')$. Let $V' = \{s\} \cup Q \cup M \cup \{t\}$, where s and t are the new source and sink node, and $E' = \{(s, q_i) : q_i \in Q\} \cup \{(q_i, vnf_j) : q_i \in Q, vnf_j \in M\} \cup \{(vnf_j, t) : vnf_j \in M\}$. For each edge (s, q_i), set its capacity as 1 and cost as 0. For each edge (vnf_j, t), set its capacity as κ_j and cost as 0. For each edge (q_i, vnf_j), set its capacity as 1 and cost as $\tau(i, j)$. Finally, we set the supply at s and the demand at t as l, indicating that there could be l VM flow migrations. This conversion technique and MCF-based solution are similar to those used in [6].

Theorem 1: The FMDV is equivalent to MCF in G'.
Proof: We show that by applying MCF upon $G'(V', E')$, it achieves all three requirements needed for optimal VM flow migration. First, each of the l VM flows is migrated to exactly one VNF instance. As the amount of supply at s is l while the capacity of each edge (s, q_i) is one, a valid flow of l amount from s to t must have one amount on each edge (s, q_i), $1 \leq i \leq l$. Due to flow conservation at any node q_i, one amount of flow thus comes out of q_i and goes into exactly one of the VNF instances vnf_j. This results in each VM flow is migrated to exactly one VNF instance. Second, as the edge capacity of (vnf_j, t) is κ_j, it guarantees that vnf_j does not take more than κ_j VM flows. Third, such a migration scheme achieves the minimum total communication and migration cost for all the l VM flows. As the cost on edge (q_i, vnf_j) is $\tau(i, j)$, the total communication and migration cost of flow q_i when it is migrated to VNF vnf_j, applying MCF algorithm upon $G'(V', E')$ thus gives the minimum total cost. ∎

B. *Benefit-based Heuristic Algorithms.*

We first introduce a key concept used in our algorithms.

Definition 1: (**Benefit of Flow Migration.**) Given a VDC graph $G(V, E)$ and initial flow assignment $p(i)$, the benefit of migrating q_i from $vnf_{p(i)}$ to another VNF vnf_j, denoted as $\mathcal{B}_i^j$, is the total cost reduction resulted from this flow migration. That is, $\mathcal{B}_i^j$ equals q_i's communication cost reduction due to its migration minus its incurred flow migration cost. $\mathcal{B}_i^j = c_{i,p(i)} - c_{i,j} - \mu \cdot c(w(p(i)), w(j))$. □

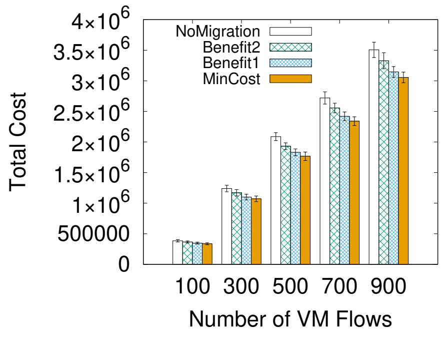

Fig. 3. Varying number of VM flows l; $k = 8$, $m = 5$, and $\mu = 100$.

Benefit-based Algorithm 1. Algo. 1 works as follows. First, we sort all the VM flows in the non-ascending order of their traffic rates. Then, we migrate each VM flow to a VNF that gives the largest benefit without violating this VNF's capacity. This continues until all the VM flows are migrated. Note that as the VM flows are initially assigned to the VNFs thus VNFs could be in their full capacity, we assume that the VNFs are initially empty in order to migrate the flows. However, this does not affect the correctness of our algorithm as finally each flow is migrated to only one VNF and the number of flows migrated to any VNF instance does not exceed its processing capacity. Finding the minimum communication cost between of all flows take $O(l \cdot |V|^2 \cdot \log|V|)$. Migrating VM flows to VNF instances takes $O(l \cdot m)$, where m is bounded by $|V|$. Therefore the time complexity of Algo. 1 is $O(l \cdot |V|^2 \cdot \log|V|)$.

```
Algorithm 1: Benefit-Based Algorithm 1.
Input: A VDC $G(V, E)$ with $l$ VM flows and $m$ VNFs.
Output: Flow migration scheme $f(i)$ that $q_i$ is migrated
    from $vnf_{p(i)}$ to $vnf_{f(i)}$, and the resulted total
    communication and flow migration cost $C_t^f$.
Notations: $p(i)$: $q_i$'s initial assigned VNF $vnf_{p(i)}$;
    $f(i)$: $(v_i, v_i')$ migrates to VNF $vnf_{f(i)}$;
    $load_j$: the current load of $vnf_j$, initially 0;
    $\mathcal{B}_i$: the largest benefit of migrating $q_i$, initially $-\infty$;
1.   Compute $C_c^p$, the total communication cost under initial
     VNF assignment $p(i)$;
2.   Sort all VM flows in the non-ascending order of their
     traffic rates. WLOG, assume $\lambda_1 \geq \lambda_2, ..., \geq \lambda_l$;
3.   for ($i = 1$ to $l$)
4.       $\mathcal{B}_i = -\infty$;
5.       for ($j = 1$ to $m$)
6.           if ($load_j < \kappa_j$)
7.               $\mathcal{B}_i^j = c_{i,p(i)} - c_{i,j} - \mu \cdot c(w(p(i)), w(j))$;
8.               if ($\mathcal{B}_i^j > \mathcal{B}_i$)
9.                   $\mathcal{B}_i = \mathcal{B}_i^j$, $f(i) = j$;
10.              end if;
11.          end if;
```

```
12.         end for;
13.         $C_c^p = C_c^p - \mathcal{B}_i$;
14.         $load_{f(i)}$++;
15.     end for;
16.     $C_t^f = C_c^p$;
17.     RETURN $\{f(1), f(2), ...f(m)\}$ and $C_t^f$.
```

Benefit-Based Algorithm 2. In Algo. 2, for VNF instance vnf_j, we migrate κ_j VM flows to vnf_j that have not been migrated and that those migrations give the top κ_j maximum benefits when visiting vnf_j. The running time of Algo. 2 is again $O(l \cdot |V|^2 \cdot \log|V|)$.

```
Algorithm 2: Benefit-Based Algorithm 2.
Input: A VDC $G(V,E)$ with $l$ VM flows and $m$ VNFs
Output: Flow migration scheme $f(i)$ that $q_i$ is migrated
    from $vnf_{p(i)}$ to $vnf_{f(i)}$, and total communication
    and flow migration cost $C_t^f$.
Notations: $p(i)$: $q_i$'s initial assigned VNF $vnf_{p(i)}$;
    $f(i)$: $(v_i, v_i')$ migrates to VNF $vnf_{f(i)}$;
    $\mathcal{F}_j$: the set of VM flows migrated to $vnf_j$;
    $migrated_i$: true if $q_i$ has already migrated, initially false;
1.  Compute $C_c^p$, the total communication cost under initial
    VNF assignment $p(i)$;
2.  for ($j = 1$ to $m$)
3.      $\mathcal{F}_j = \phi$;
4.      for ($i = 1$ to $l$)
5.          if ($migrated_i == false$)
6.              $\mathcal{B}_i^j = c_{i,p(i)} - c_{i,j} - \mu \cdot c(w(p(i)), w(j))$;
7.              $\mathcal{F}_j = \{(i, \mathcal{B}_i^j)\} \cup \mathcal{F}_j$;
8.          end if;
9.      end for;
10.     Sort $\mathcal{F}_j$ in the non-ascending order of $\mathcal{B}_i^j$;
11.     $\mathcal{F}_j = \{(x_1, \mathcal{B}_{x_1}^j), (x_2, \mathcal{B}_{x_2}^j), ...\}$, where $\mathcal{B}_{x_1}^j \geq \mathcal{B}_{x_2}^j ...$;
12.     for ($k = 1$ to $\kappa_j$)
13.         $C_c^p = C_c^p - \mathcal{B}_{x_k}^j$;
14.         $f(x_k) = j$;
15.         $migrated_{x_k} = true$;
16.     end for;
17. end for;
18. $C_t^f = C_c^p$;
19. RETURN $\{f(1), f(2), ...f(m)\}$ and $C_t^f$.
```

V. Performance Evaluation

Simulation Setting. We investigate the performances of benefit-based algorithms viz. Algo. 1 and Algo. 2 (referred to as **Benefit1** and **Benefit2**), and minimum cost flow algorithm (referred to as **MinCost**). We compare their total costs of VM flow migration and communication with the case without flow migration, which is referred to as **NoMigration**. We consider small $k = 8$ VDCs with 128 PMs and large $k = 16$ VDCs with 1024 PMs. The VMs are randomly placed on the PMs and the VNF instances are randomly placed on the switches. Each data point in all the plots is an average of 10 runs, and the error bars indicate 95% of the confidence interval. For the small $k = 8$ VDC, we set the VNF processing capacity κ_j as $\lceil \frac{l}{m} \rceil$, a most stressful flow migration scenario wherein each VNF is operating around its full processing capacity. For the large $k = 16$ VDC, we set κ_j as $2 \times \lceil \frac{l}{m} \rceil$ as more resources are available in large VDCs.

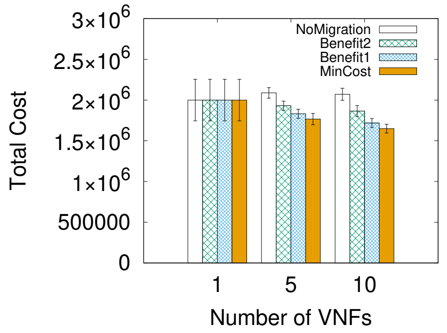

Fig. 4. Varying number of VNF instances m; $k = 8$, $l = 500$, and $\mu = 100$.

Varying Number of VM Flows l. Fig. 3 varies l from 100, 300, ..., to 900 while fixing the number of VNF instances m as 5 and the flow migration coefficient μ as 100. The traffic rates of all the VM flows are random numbers in [0, 1000]. The total cost of each algorithm increases with the increase of l, as more VM flows in general incur more network traffic. We also observe that all the three algorithms viz. Benefit1, Benefit2, and MinCost yield less total network cost compared to NoMigration, demonstrating that our VM flow migration algorithms are effective in reducing network traffic.

Varying Number of VNF Instances m. Fig. 4 investigates the effect of m on the total network cost. We have several observations. First, when there is only one VNF instance, the three algorithms perform the same as the NoMigration, as all the VM flows must visit the same VNF independent of the algorithms. Second, when $m = 5$ and 10, our three algorithms perform better than NoMigration by yielding lower network cost, with MinCost performing the best. With the increase of m, the performance improvement of our algorithms upon NoMigration increases, showing that our algorithms are indeed effective in reducing network traffic. Finally, the total costs of all algorithms decrease with the increase of m. Although the VNFs are operating in their full processing capacity in all cases, as VNFs are randomly placed into the network, there are more VNFs (with fewer capacities) in the network when m is large. This gives VM flows more options to choose which VNFs to migrate to in order to reduce the traffic costs.

Dynamic Cloud Traffic in VDCs. Next, we investigate how our algorithms perform in a dynamic scenario where the traffic rates of VM flows are constantly changing. As MinCost works best among the three algorithms, we compare MinCost with NoMigration for 10 epochs. At the beginning of each epoch, all the VM flows randomly change their traffic rates to new values in [0,1000]. Then, MinCost executes and finds an optimal flow migration scheme while NoMigration simply recalculates the total communication cost of all the VM flows. We set migration coefficient μ as 100 and 200. Fig. 5 shows

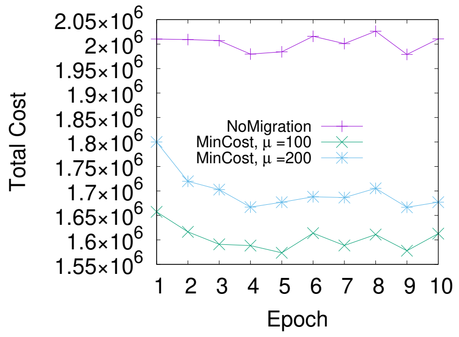

Fig. 5. Dynamic network traffic in 10 epochs; $k = 8$, $l = 500$, $m = 5$.

that MinCost outperforms NoMigration constantly in all the epochs. Further, it is able to reduce more network costs at $\mu = 100$ compared to $\mu = 200$, due to smaller overhead cost of flow migration at $\mu = 100$. MinCost is able to reduce network cost by around 20% compared to NoMigration.

Performances in Large-Scale VDCs. Finally, Fig. 6 investigates the performances of our algorithms in large-scale VDCs with large numbers of PMs, VM flows, and VNFs. We consider $k = 16$ VDCs of 1024 PMs and vary l from 1000 to 3000 while fixing m as 20 and μ as 100. It shows that all our three designed algorithms are able to reduce network traffic compared to NoMigration. In particular, the MinCost reduces up to 28% of network cost compared to NoMigration. This shows that our VM flow migration algorithms achieve efficient traffic mitigation in large-scale VDCs as well.

VI. Conclusion and Future Work

We proposed a new flow migration problem called FMDV in dynamic and VNF-enabled cloud data centers. The goal of FMDV is to optimize network resources such as bandwidth and energy consumption by reducing the network traffic via VM flows migration among different VNFs. We proposed an optimal and efficient minimum cost flow-based algorithm and two benefit-based efficient heuristics to solve the FMDV. We showed via extensive simulations that they are all effective traffic-mitigation techniques, reducing the network traffic by up to 28% compared to the case without flow migration. For future work, we will consider SFCs wherein VM traffic must traverse a chain of VNFs with different functions. We will also consider that the processing capacities of VNFs are in terms of total traffic rates they can process, not the number of VM

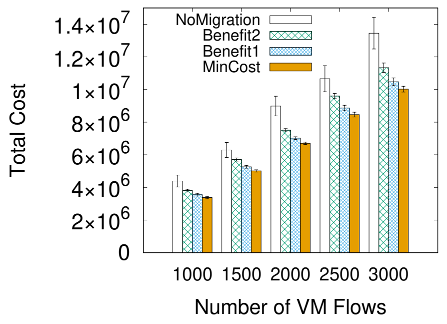

Fig. 6. Performances in large-scale VDCs of $k = 16$, $m = 20$, $\mu = 100$.

flows assumed in the current work. How to design an efficient flow migration scheme to minimize network traffic becomes a new and challenging problem.

Acknowledgment

This work was supported in part by NSF Grants 1911191, 1923956, 2034030, and 2036143.

References

[1] Zoom cloud meetings. https://zoom.us/.
[2] Zoom meeting connector core concepts. https://support.zoom.us/hc/en-us/articles/201363113-Meeting-Connector-Core-Concepts.
[3] P. Aguilera, C. Gonzalez, and B. Tang. Achieving virtual network function load-balanced flow migration in dynamic cloud data centers centers. In *Proc. of the First Computer Science Conference for CSU Undergraduates (CSCSU 2021)*.
[4] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice-Hall, Inc., 1993.
[5] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable, commodity data center network architecture. *SIGCOMM Comput. Commun. Rev.*, 38(4):63–74, 2008.
[6] M. Alqarni, A. Ing, and B. Tang. Lb-map: Load-balanced middlebox assignment in policy-driven data centers. In *Proc. of IEEE ICCCN 2017*.
[7] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. A view of cloud computing. *Commun. ACM*, 53(4):50–58, 2010.
[8] L. Cui, F. P. Tso, D. P. Pezaros, W. Jia, and W. Zhao. Plan: Joint policy- and network-aware vm management for cloud data centers. *IEEE Transactions on Parallel and Dis. Sys.*, 28(4):1163–1175, 2017.
[9] A. Jukan F. Carpio, S. Dhahri. Vnf placement with replication for load balancing in nfv networks. In *Proc. of IEEE ICC 2017*.
[10] X. Fei, F. Liu, H. Xu, and H. Jin. Adaptive vnf scaling and flow routing with proactive demand prediction. In *Proc. of IEEE INFOCOM 2018*.
[11] H. Flores, V. Tran, and B. Tang. Pam & pal: Policy-aware virtual machine migration and placement in dynamic cloud data centers. In *Proc. of IEEE INFOCOM 2020*.
[12] A. V. Goldberg. An efficient implementation of a scaling minimum-cost flow algorithm. *J. Algorithms*, 22:1–29, 1997.
[13] P. Khani, B. Tang, J. Han, and M. Beheshti. Power-efficient virtual machine replication in data centers. In *Proc. of IEEE ICC 2016*.
[14] R. J. Martins, C. B. Both, J. A. Wickboldt, and L. Z. Granville. Virtual network functions migration cost: from identification to prediction. *Computer Networks*, 181(9), 2020.
[15] 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 Sur. and Tut.*, 18(1), 2015.
[16] Z. A. Qazi, C.-C. Tu, L. Chiang, R. Miao, V. Sekar, and M. Yu. Simplefying middlebox policy enforcement using sdn. In *Proc. of ACM SIGCOMM 2013*.
[17] K. Qu, W. Zhuang, Q. Ye, X. Shen, X. Li, and J. Rao. Dynamic flow migration for embedded services in sdn/nfv-enabled 5g core networks. *IEEE Transactions on Communications*, 68(4):2394–2408, 2020.
[18] A. Roy, H. Zeng, J. Bagga, G. Porter, and A. C. Snoeren. Inside the social network's (datacenter) network. In *Proc. of SIGCOMM 2015*.
[19] C. Sun, J. Bi, Z. Meng, T. Yang, X. Zhang, and H. Hu. Enabling nfv elasticity control with optimized flow migration. *IEEE Journal on Selected Areas in Communications*, 36(10):2288–2303, 2018.
[20] B. Tang, N. Jaggi, and M. Takahashi. Achieving data k-availability in intermittently connected sensor networks. In *Proc. of the International Conference on Computer Communications and Networks (ICCCN'14)*.
[21] L. Tang, X. He, P. Zhao, G. Zhao, Y. Zhou, and Q. Chen. Virtual network function migration based on dynamic resource requirements prediction. *IEEE Access*, 7:112348–112362, 2019.
[22] Y. Wang, G. Xie, Z. Li, P. He, and K. Salamatian. Transparent flow migration for nfv. In *Proc. of the ICNP*, 2016.
[23] X. Zhang, C. Wu, Z. Li, and F. C.M. Lau. Proactive vnf provisioning with multi-timescale cloud resources: Fusing online learning and online optimization. In *Proc. of IEEE INFOCOM 2017*.
[24] 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 IEEE ICNP 2013*.