Latency and Policy Aware Hierarchical Partitioning for NFV Systems

Dilip Krishnaswamy, Ravi Kothari, Vijay Gabale
IBM Research

***Abstract*— This paper explores latency-aware and policy-aware optimized placement of virtual network functions across data centers in NFV systems. A hierarchy of distributed data centers is suggested to support network function software appliances with the flexibility to place functions based on performance requirements in the hierarchy, and different test scenarios for optimal VNF placement are studied. The paper discusses options for distributed function virtualization such as hierarchical partitioning, collapsing, replication, and north/south function splitting, to explore VNF placement options in NFV systems.**

I. Introduction

Networks are evolving towards the virtualization of network functions, and network functions are being designed to execute in virtual machines (VMs), with such functions commonly being re-allocated to execute in a cloud data center (DC). In general, there is an emerging trend to migrate dedicated hardware network appliances to virtualized software appliances (VMs or containers) running on data centers to reduce capital and operating expenditures for network operators. However, to meet performance requirements for network protocol stacks, it would be desirable to have in-network data centers in addition to cloud data centers to support such network functions virtualization (NFV). For this purpose, data centers will have to move into networks to augment the cloud. These in-network data centers will provide compute and storage capacity internal to the network to help deploy network functions and services. This naturally leads to network architectures which will include a hierarchy of data centers to enable NFV. The paper describes possibilities utilizing hierarchical NFV data centers to enable the deployment of NFV systems with latency and policy awareness. Traditionally, NFV architecture includes various components for managing resources such as the NFVI, the VIM, and the VNF Manager. The NFV Infrastructure (NFVI) manages the physical compute/storage/networking resources. The Virtual Infrastructure Manager (VIM) performs lifecycle management of these available infrastructure resources, maintains a dynamic resource pool of the same. The VNF (Virtualized Network Function) Manager manages each of the individual VNFs in the system, and exchanges information on virtual vs physical resource mapping with the VIM. An Orchestrator determines the mapping of the requirements in the system to the different available resources in the system. The work in this paper is targeted towards enabling latency and policy aware partitioning / orchestration in the system to map VNFs across available VIM resources across data centers in a NFV system, to meet performance requirements in the system. Additionally, the paper address variations for hierarchical function partitioning and collapsing in such systems. With the availability of both cloud and in-network data centers, VMs associated with network, service or application functions can be supported across data centers, enabling distributed function virtualization (DFV) (Figure 2) across such data centers.

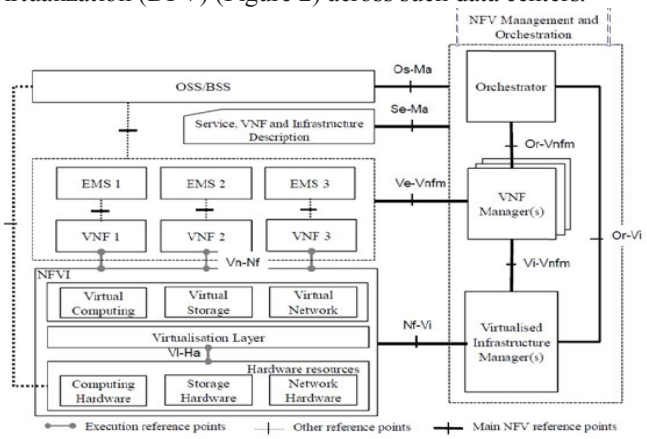

Figure 1: The ETSI NFV Architectural Framework [1]

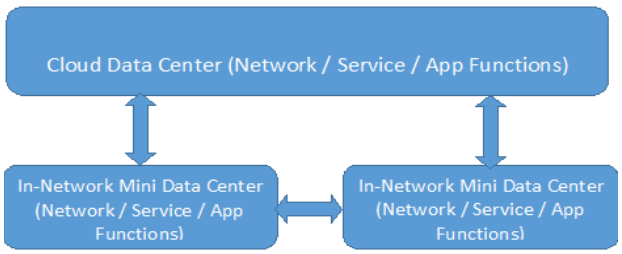

Figure 2: Distributed Function Virtualization (DFV)

II. Recent Progress In NFV Research

Excellent work is in progress in the NFV space in ETSI-NFV with discussions on architectures and use-cases and network changes required to support this evolution to NFV [1]. Researchers have started exploring the possibility virtualizing networks [2], and virtualizing base-stations [3] as well in LTE-Advanced networks. In the work in [4], the authors discuss cellular service provider owned carrier clouds and cover several aspects of carrier clouds such as software-defined networking inside the cloud and the resource provisioning. In [5], the authors describe an architecture based on an orchestrator that automatically places virtual nodes and services supported by a monitoring system that manages resource usage. In [6], the authors suggest a new network interface based on network function virtualization where limited access to networks can be given to third-party developers such as video on-demand service provider requiring content caching in access networks. Research work in [7] and [8] explores edge deployments to bring functions and services

978-1-4673-6884-1/15/$31.00 ©2015 IEEE

closer to the users. Recent work has begun in the network functions virtualization research group (nfvrg) [9] in the Internet Research Task Force, where the key recent areas of interest relating to policy-based resource management, service verification, analytics, and security have emerged. However, hierarchical latency and policy aware VNF partitioning has not yet been explored in the literature. In general, there is significant activity in both industry and academia to further the state of the art in architecting and designing NFV systems. This paper will explore partitioning NFV systems hierarchically based on latency and policy constraints in such systems.

III. Hierarchical Function Virtualization

NFV data centers are placed in a hierarchy based on their differential abilities to serve a set of users such as, for example, mobile devices and users in a given geographical region, or enterprise users in a given location, or a community of users in an area with limited or poor connectivity. Data centers closer to users can provide greater sensitivity with regard to delivering function performance (such as faster responses) compared to data centers that are farther away from users/devices in the NFV data center hierarchy. Such a hierarchical network of data centers, can support a distributed partitioning of network functions VMs. In addition, based on resource availability in these data centers, additional service and/or application function VMs could also be supported. Here network functions can include functions provided by a traditional hardware appliance such as a services gateway (S-GW) or a packet gateway (P-GW) in a Long Term Evolution (LTE) network. Network functions can also including functions such as a load balancer or a firewall implementation in a network. Service functions can include, for example, a combination of functions that help with providing a specific service such as a set of functions that can combine to provide resources within a cellular network for access (for instance, providing resources in the access network and a core network to enable access), a function to provide an authentication service in a network, and/or a service function enabling a delay-tolerant content delivery service. Additional specialized service functions can also be considered such as an inter-operator tunneling service function, an application proxy VM, or a machine-to-machine (M2M) service layer VM. Application function VMs can include support for different applications such as an augmented reality application, an email application, a social networking application, and over-the-top audio/video applications.

IV. Hierarchical Function Partitioning

The functions that need to be supported in an NFV system will need to be partitioned across the hierarchical network based on (i) a desired performance sensitivity measure (a latency requirement) associated with each of the functions and (ii) data center sensitivity measures (data center latency to serve functions). An example of hierarchical function partitioning to serve a UE (User Equipment) in a cellular network is shown in Figure 3. Latency-sensitive functions related to a radio network controller (RNC), an Evolved Node B (eNodeB), a serving general packet radio service support node (SGSN), a mobile management entity (MME), or a serving gateway (S-GW), can be mapped to a lower data center (LDC).

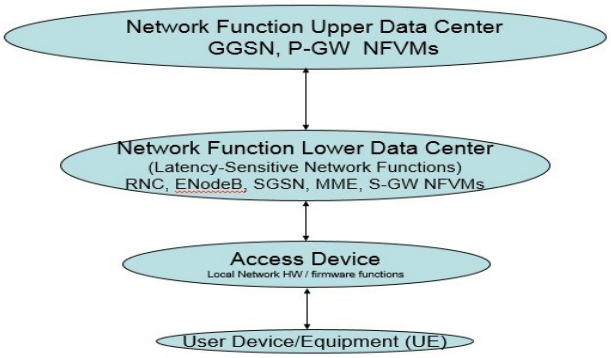

Figure 3: Typical NFV Data Center Hierarchy for a UE

Here, highly latency sensitive functions can execute in hardware in traditional hardware appliances to serve lower-level NodeB/RNC/eNodeB functions (such as the physical (PHY) layer, the media access control (MAC) layer, and/or the radio link control (RLC) layer), whereas upper layers such as the PDCP layer, IP-packet fragmentation or defragmentation, or programmable connectivity to SGSNs or S-GWs (hardware appliances or virtual software appliances) can be enabled via software appliance implementations using RNC VMs or eNodeB VMs in the LDC. Less latency-sensitive Network Function VMs (NFVMs) (for example, VMs, that handle functions related to a gateway GPRS support node (GGSN), a packet data network gateway (P-GW), etc.) can be mapped to an upper data center (UDC). Figure 4 shows how such a partitioned hierarchy can coexist with existing hardware appliances (on the bottom and middle-right of the figure), so that the software appliances for the different network components can be instantiated and utilized as needed. eNodeBs can connect to a traditional hardware appliance-based S-GW or to an S-GW VNF. Similarly, either a hardware appliance version of the S-GW or an S-GW VNF can connect with a hardware appliance version of a P-GW or a P-GW VNF as well. Eventually, one can anticipate that the hardware appliances will be replaced with software appliances. However, in the near term, both can co-exist, and it becomes desirable to place the software appliances correctly in the network hierarchy to allow existing networks to scale and support more users and sessions dynamically. To place the functions, one can consider data centers at different levels in a latency-aware hierarchy. For users at a given geographic location, different data centers can serve them differently. Relative to the users and the end-to-end latency between a data center and the user devices, a latency-aware hierarchy is naturally imposed in the system, such that the data centers closest to the users (L1DC data centers) provide the lowest mean latencies for interaction with user devices. Data centers further away can be arranged in hierarchical order such as L2DC data centers and L3DC data centers followed by CDCs (cloud data centers) with increasing latencies for interaction with user devices. An example of hierarchically split network functions across data centers for LTE is shown in Fig 5 (left).

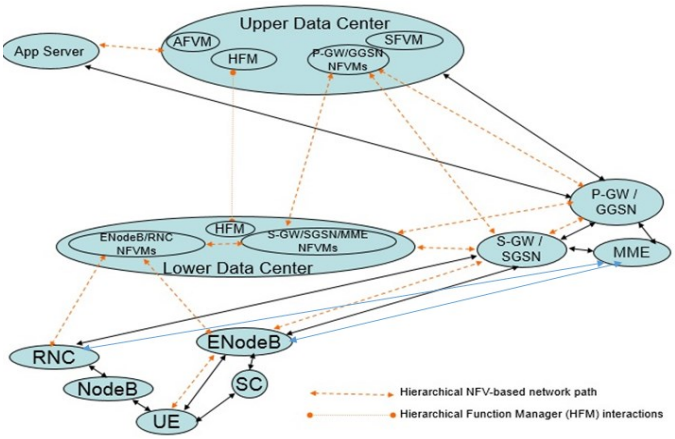

Figure 4: Coexistence with existing hardware appliances

A. *Hierarchical Function Collapsing (HFC)*

With the availability of software appliance functions for different components in the network, one can also consider hierarchical collapsing of functions (such as collapsing the set of network functions that relate to an overall service function for 3G or 4G cellular services) into an L1DC (example of a fully collapsed hierarchical network is shown in Figure 5 (right)), or collapsed partitioning of functions between a L1DC and an L2DC (an example of a partially collapsed hierarchical network is shown in Figure 5 (middle)). Alternatively or additionally, application or service functions VMs from the cloud can be collapsed to an in-network DC. This can enable application or service functions to execute in the L1DC or L2DC or L3DC, to deliver improved performance in the system. Collapsing functions can be performed to reduce inter-function communications costs, to improve application performance associated with applications that are supported over the network, and/or to provide differentiated services for a given set of users that have a higher quality-of-service requirement or cost-of-service requirement compared to another set of users. Quality-of-service requirements and/or performance requirements can pertain, for example, to latency, bandwidth, jitter, and delay tolerance associated with application flows.

Examples of such application functions can include delivery of content stored at an L1DC, L2DC, or L3DC, or a CDC (cloud data center) supporting a mobile gaming application or an augmented reality application at any of the data centers, supporting social networking of users using a social networking application function, and/or supporting internet access via these data centers. Network functions such as firewalls, load balancers or security/authentication/usage-monitoring functions can be replicated across these data centers as needed. The supported applications can be placed in one of the data centers in the hierarchy or can be replicated across the data centers. Depending on the needs of the users and/or the service level agreements associated with the service for such applications, different users can be mapped to application functions residing at different locations in the data center hierarchy. Collapsed function systems can also provide a social benefit in scenarios where connectivity to the cloud may be limited (for example in rural areas), to help establish local communication paths, or to enabling local applications to execute end-to-end between users or between a user and a local server, to enable communication paths at the edge.

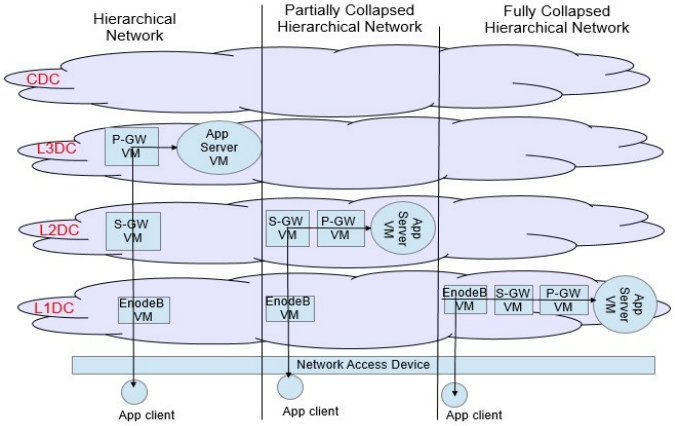

Figure 5: Hierarchical Function Collapsing (HFC)

V. Latency And Policy Aware Partitioning

In this section we describe a methodology to place VNFs in an NFV data center hierarchy while meeting latency and policy constraints in the NFV system.

A. *Policy-awareness considerations*

Let us assume that a set of data centers D will be used to serve users in a given geographic location. Data centers need to be policy-aware during dynamic operation. Policies can take different forms such as determining which data centers can operate at a given location and time, or specifying the desired performance requirements for network functions in the system. Let the vector $\boldsymbol{p(t)}$ represent a policy constraint that represents whether a data center will support users in region G at a given time. Therefore, each component

$$p_i(t) \varepsilon \{0,1\} \quad \forall\, i \qquad \text{……..(1)}$$

in the vector $\boldsymbol{p(t)}$ represents the policy constraint whether a data center is operational or not. The policy can be time-varying and dynamic. Some data centers may become non-operational during periods of low load for example, or may become unavailable for new VNF allocation due to heavy loading. Therefore the policy can be periodically checked, and the placement of functions can be re-visited whenever the policy changes. A policy vector can also be used to specify the latency constraints associated with the network functions, or to specify the relative priorities of VNFs or relative costs of data centers, which are described in the subsequent sub-sections.

B. *Latency-awareness considerations*

Let us assume that there are N network functions that need to be supported. Let us assume that M_{ij} is the latency associated with the processing of a network function j (where $j \varepsilon N$) on a data center $i \varepsilon D$ that serves users in a geographic region G. If a data center $i \varepsilon D$ has an inherent latency α_i to serve users in region G due to network delays and internal load, and the processing delay associated with the VNF at that data center

incurs a delay τ_j , then M_{ij} is given by $M_{ij} = \alpha_i + \tau_j$. If $\tau_j << \alpha_i$, then, $M_{ij} \approx \alpha_i$. The set of allowable latencies for the N network functions can be specified by a policy vector $\underline{\boldsymbol{\Lambda}}$ where $\lambda_j \ \varepsilon \ \underline{\boldsymbol{\Lambda}}$. Here λ_j is the maximum allowable latency for the execution of a network function j (where $j \ \varepsilon \ N$) relative to the users in the geographical region G. This results in a set of latency-awareness constraints that a data center i needs to satisfy to allow a network function j to be processed at that data center, relative to the users in geographic region G.

$$M_{ij} \leq \lambda_j \qquad \forall \text{ i,j} \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

C. VNF mapping to data centers

Let us assume that κ_{ij} represents the assignment of a network function j to a data center i (1 if mapped, 0 otherwise) subject to the constraints given below.

$$\kappa_{ij} \ \varepsilon \ \{0,1\} \qquad \forall \ i,j \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots(3a)$$

$$\Sigma_i \ \kappa_{ij} \ = 1 \qquad \forall \ j \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots(3b)$$

and also adhering to the operational policy constraint to obtain

$$\kappa_{ij} \leq p_i \qquad \forall \ i,j \ \ldots\ldots\ldots\ldots\ldots\ldots\ldots(3c)$$

Equation (3b) ensures that each network function is mapped to a single data center. Equation (3c) ensures that a network function is not allocated to a data center that is not actively supporting the VNFs at a given time. This results in an N x D matrix $\boldsymbol{K}$ comprising elements κ_{ij} to specify the mapping of VNFs to Data Centers.

D. Relative Priorities for VNFs

Let β_j represent the relative priorities of VNF j (where $j \ \varepsilon \ N$) so that a particular VNF can be given higher priority relative to another VNF (if desired) to operate at a particular data center. For example, if an MME VNF that handles user mobility needs to be operated at a more latency sensitive data center, then the MME VNF can be given higher priority over a P-GW VNF that may have relatively more relaxed latency constraints.

E. Relative Data Center Costs

Each data center may have limited resources so that the cost of utilizing the resources in the data center may vary. Also, depending on the location of the data center, the energy required to operate the data center can vary. For example, an in-network L1DC data center in a city is likely to incur higher costs of operation per kWH, relative to an L3DC data center in a remote location powered by a solar panel array farm. Let us assume that C_i represents the cost of operating and utilizing resources of a data center i. Depending on the dynamic electrical energy utility costs or limited resource availability or data center availability constraints, the cost of allocation of a function to a data center can vary, and can be utilized in a utility function to determine placement of the VMs. For purposes of optimization, a relative data center cost measure or ratio across data centers is adequate. If a data center needs to be phased out of operation due to a policy constraint, its relative cost can be dynamically increased to divert resource allocation away from the data center.

F. Latency Slack Availability

The actual difference in latencies between the desired performance requirement for the function (λ_j) and the available performance from the data center (M_{ij}) can be utilized in a utility function to reflect the improvement in performance with respect to latency considerations. For this purpose, let us define a latency slack difference measure

$$\begin{aligned} L_{ij} &= \lambda_j - M_{ij} && \text{if } M_{ij} < (\lambda_j - \delta) \\ &= \delta && \text{if } M_{ij} \geq (\lambda_j - \delta) \end{aligned} \qquad \ldots\ldots\ldots(4)$$

where δ is a small threshold value for the latency.

The higher the latency slack difference measure, the better is the performance delivered by a data center i for a network function j when that network function is mapped to the data center. For the utility function defined subsequently, if the latency constraint is not met, then then latency difference measure is set to a very small value (say δ = 0.0001 or a "min_float" value) to allow the utility function to be evaluated.

G. Optimized Placement

Based on the knowledge of these constraints, a system-wide goal would need to be met be to minimize a utility function to place the network functions in the data center hierarchy. A system-wide utility function U can be defined given by

$$U \ = \ \Sigma_{i,j} \ \beta_j \ * \ (C_i / L_{ij}) \ * \ \kappa_{ij} \qquad \ldots\ldots\ldots\ldots(5)$$

For a given mapping (κ_{ij}) of network functions to data centers, this utility function combines the relative priorities of the VNFs, the costs of operating the data centers, and the performance of the VNFs. The objective of the optimization problem can then be defined to be

Minimize

$$U \ = \ \Sigma_{i,j} \ \beta_j \ * \ (C_i / L_{ij}) \ * \ \kappa_{ij}$$

Subject to the constraints

$$\begin{aligned} & p_i(t) \quad \varepsilon \ \{0,1\} && \forall \ i \\ & M_i < \lambda_j && \forall \ i,j \\ & \kappa_{ij} \ \varepsilon \ \{0,1\} && \forall \ i,j \\ & \Sigma_i \ \kappa_{ij} \ = 1 && \forall \ j \\ & \kappa_{ij} \leq p_i && \forall \ i,j \end{aligned}$$

Minimizing the utility function (5) subject to the constraints (1), (2), (3a), (3b) and (3c), results in an optimal partitioning for the VNFs in the data center hierarchy. The optimization problem was solved using the ILOG CPLEX optimization library [10]. The output of the optimization is an allocation matrix of values $\boldsymbol{K}$ mapping VNFs to data centers.

H. Optimized Placement Test Scenarios

Consider an example of 3 data centers serving users in a given region (such as Cupertino, CA, USA) where the set of three data centers D consists of data-centers D_1, D_2, and D_3 (for example located in Cupertino, nearby Sunnyvale,, and farther-away San Francisco) respectively. Let us assume that the task is to place VNFs related to the eNodeB, S-GW, MME, and the P-GW functionalities, which are represented by the functions N_1, N_2, N_3 and N_4, respectively.

1) Test Scenario 1

Let us assume that the latency (in ms) vector $\underline{\boldsymbol{\Lambda}}$ = [15, 30, 25, 80] for the 4 VNFs. Let us assume that the N x D matrix **M** (for M_{ij} values in ms) = [[10 20 70] [10 20 70] [10 20 70] [10 20 70]] for each of the 4 functions relative to the 3 data centers resulting in a latency slack difference measure matrix **L** = [[5 δ δ] [20 10 δ] [15 5 δ] [70 60 10]]. Let the policy vector **p(t)**

= [1 1 1] (implying that all data centers available to support the VNFs). If equal relative costs of operation (C_i) and equal priorities of the VNFs (β_j) are assumed, then solving the optimization problem results in an allocation of all the VNFs to data center D_1. The N x D allocation matrix of the VNFs to the data centers is given by **K** = [[1 0 0] [1 0 0] [1 0 0] [1 0 0]]. It is natural to expect that if all the data centers are equally expensive, then the system will prefer a VNF allocation that delivers the highest performance for the VNFs by placing the VNFs in data center D_1.

2) Test Scenario 2

The system is identical to Test Scenario 1 except that the cost of utilization of data center D_1 is increased relative to D_2, and D_3 in the ratio 100:1:1. This results in a partially collapsed mapping of functions where function N_1 is mapped to data center D_1, whereas the remaining functions N_2, N_3, and N_4 map to data center D_2. The allocation matrix is given by **K** = [[1 0 0] [0 1 0] [0 1 0] [0 1 0]]. This implies that increasing the cost of a data center (in this case D_1) results in the placement of VNFs at a lower cost data center D_2 while meeting latency constraints. This enables better utilization of the data centers, such that the VNFs that are have more relaxed latency requirements are managed by alternate data centers that have a lower cost or a higher latency to provide service.

3) Test Scenario 3

The system is identical to Test Scenario 1 except that the relative cost of utilization of the data centers C_i is in the ratio 100:10:1. In this case, the functions are mapped such that function N_1 is mapped to data center D_1, functions N_2, and N_3 map to data center D_2, and function N_4 maps to data center D_3. The allocation matrix is given by **K** = [[1 0 0] [0 1 0] [0 1 0] [0 0 1]]. Relative to Test Scenario 2, in this case, the data center D_2 is more expensive to use relative to data center D_3, so that the least latency sensitive VNF N_4 (the P-GW VNF) is positioned in data center D_3.

4) Test Scenario 4

In this test scenario, all constraints as kept the same as in Test Scenario 3, except for a change in the policy vector. If we impose a dynamic operational policy constraint such that the policy vector **p(t)** = [1 1 0] , then function N_1 is mapped to data center D_1, whereas functions N_2, N_3, and N_4 map to data center D_2. The allocation matrix is given by **K** = [[1 0 0] [0 1 0] [0 1 0] [0 1 0]]. Due to the non-availability of data center D_3, the P-GW VM is placed at data center D_2.

5) Test Scenario 5

Relative to test scenario 3, if the policy constraint changes to **p(t)** = [1 0 1], then functions N_2, and N_3, are placed at data center D_1 along with function N_1. Function N_4 is mapped to data center D_3. The allocation matrix is given by **K** = [[1 0 0] [1 0 0] [1 0 0] [0 0 1]]. Due to the non-availability of data center D_2, the latency-sensitive VNFs for the S-GW and the MME are placed at the more expensive data center D_1, to meet latency constraints.

6) Test Scenario 6

Alternatively, relative to test scenario 3, if functions are allocated a differential priority vector **β** = [10 8 8 1] for example, across the 4 VMs, then the functions N_1 , N_2, and N_3, map to data center D_1, whereas function N_4 remains on data center D_3.

7) Test Scenario 7

The latency to communicate with a data center can vary dynamically as such latency can be impacted by the network load or the data center load, so that the partitioning decisions may vary dynamically. During periods of very high load, it is possible that equation (2) is not satisfied, and under such circumstances, the partitioning decisions can be revisited. Let us assume that the data center D_2 becomes heavily loaded resulting in the N x D matrix **M** = [[10 35 70] [10 35 70] [10 35 70] [10 35 70]] for each of the 4 functions relative to the 3 data centers resulting in a latency slack difference measure matrix **L** = [[5 δ δ] [20 δ δ] [15 δ δ] [70 45 10]]. Then this forces the allocation in all scenarios of VNFs N_1, N_2, and N_3 (ENodeB, MME and S-GW VNFs) to data center D_1 regardless of data center costs or VNF priorities, as allocation of these VNFs to data centers D_2, and D_3, violates latency constraints. VNF N_4 (P-GW VNF) gets mapped to any of the data centers depending on the relative costs of the data centers. When the data centers have equal relative costs 1:1:1, VNF N_4 maps to data center D_1. When the data centers have relative costs 100:1:1, VNF N_4 maps to data center D_2. When the data centers have relative costs 100:10:1, VNF N_4 maps to data center D_3.

8) Test Scenario 8

This system is identical to Test Scenario 3, except that in anticipation of an impending policy implementation for data center D_2 to be shut down to minimize energy, or for the data center to become unavailable for future resource allocations due a high load at the data center, the relative cost of utilization of the data centers C_i is modified to a ratio 100:1000:1. When the LP is initiated with these altered cost ratios, it results in an output that forces the solution to be the equivalent of Test Scenario 5, where functions N_1, N_2, and N_3, are placed at data center D_1 and function N_4 is mapped to data center D_3. The allocation matrix is given by **K** = [[1 0 0] [1 0 0] [1 0 0] [0 0 1]]. This ensures that all future allocations are diverted away from data center D_2. In the case when the data center D_2 is experiencing high load, it can continue supporting the functions that it may be currently supporting for its current load, while effectively rejecting any new instantiations of the eNodeB, MME, S-GW, or P-GW VNFs that may be needed to help with performance scaling and supporting additional users in the network. Such new VNFs instantiations can instead be done at alternate available data centers for future user sessions, while meeting VNF performance constraints in the system.

I. Proactive Monitoring and VNF Placement

In general, depending on the dynamic latencies, costs of operation, and desired performance requirements, the allocation of VNFs to data centers can vary. It would be desirable to proactively monitor the system to be aware of the network, compute, storage, and energy constraints in the system, so that VNFs can then get instantiated at alternate locations, and new user sessions can get mapped to such VNFs. Proactive monitoring of system and network constraints or anticipating change in policy constraints can

help with proactive instantiation and "soft migration" of VNFs to appropriate alternate data centers based on the constraints.

VI. Benefits of Function Collapsing

The three different hierarchical collapsed function architecture variants in Figure 5 were studied to characterize the benefits of reduction in latency through collapsing. Mininet [11] was used to setup an experimental testbed on a laptop, where different processes were spawned as real network nodes. A video application was used on a typical LTE network, and switching modules in Mininet were used to emulate the UE, eNodeB, SGW and PGW nodes.

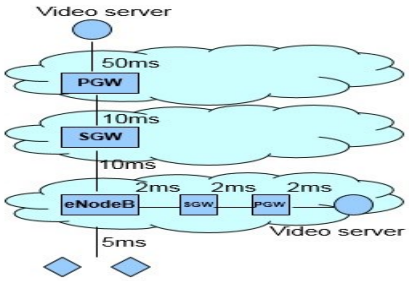

Figure 6: Simulation setup and performance

Metric	Legacy Hierarchical	Partially Collapsed	Fully Collapsed
Round trip time	179 ms (median)	64 ms (median)	22 ms (median)
Connection setup	3.7 sec (median)	1.3 sec (median)	0.7 sec (median)

Table 1: Comparison of Hierarchically Collapsed Variants

The different topology scenarios across data-centers were created as depicted in Figure 5, including latencies for communication over links between different nodes in the network. Network latency assumptions include typical mean latencies of 5ms over a cellular access link, 10ms latency over core links, and a 2ms latency for intra-data-center links (as shown in Figure 6). To study the performance impact due to variation in latencies in the networks, the latency values in Figure 6 were modulated with a uniformly random latency between 0ms and 5ms which was added to the access and core links to simulate variations in the background load. To ensure fairness, the randomness was imposed in a consistent fashion for different scenarios that were compared. To enable video flows in the environment, videos were hosted using an Apache server in a node connected to the exit node, i.e., P-GW, of the network. The following three simulation scenarios were considered as shown in Figure 5: (1) a traditional hierarchical network with the functions in different data centers, (2) a partially-collapsed network where SGW, PGW and video server is hosted inside level 2 data center (L2DC), and (3) a fully collapsed network where eNodeB, SGW, PGW and video server is hosted on level 1 data center (L1DC), closest to the users. The simulation for each of these scenarios was run 10 times for 5 different video files. Results are presented for a 70MB, 2 minute 29 second HD video clip. The average bandwidth required for the playout of this video was about 3.7Mbps. Table 1 shows the comparison of round trip time (RTT) and connection setup time for the three scenario that we consider. As the table shows, the collapsed network reduces the round-trip-time by more than 8-fold and connection setup latency by more than 5-fold, which helps in buffering video frames at a faster rate. Playout stalls for this specific setup varied typically between 12 (legacy), 2 (partially collapsed), and 0 (fully collapsed). Typically, application-level exchanges for different applications on a cellular network require multiple RTTs, so that a reduction in RTT has good benefits. It can be seen therefore that collapsing functions closer to the user and deploying services in lower level data centers can improve user quality of experience in NFV systems.

VII. Additional Variations

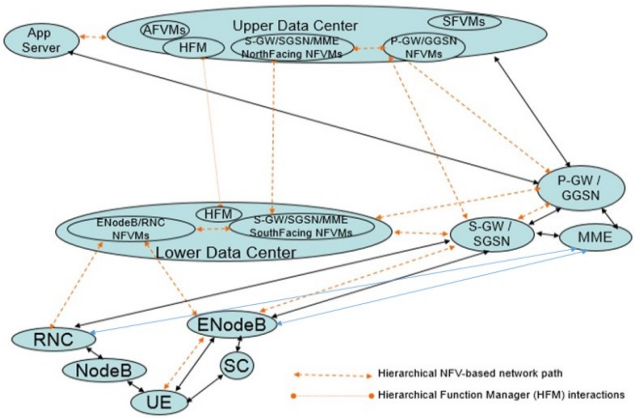

Figure 7: North/South Function Splitting

A. *Hierarchical Function Splitting (HFS)*

Different variations can be explored in the future for such hierarchical architecture particularly for emerging 5G systems. For example, intra-function optimization can be considered where south-facing functions can be placed in an LDC (lower data center), and north-facing function can be placed a UDC (upper data center), such that a VNF executing in a traditional network node (such as an S-GW or an SGSN) is now split between two data centers (Figure 7). An example of a south-facing S-GW function can include the communication of buffered data from an S-GW to a target eNodeB to enable a hand-off from a serving eNodeB to a target eNodeB. An example of a north-facing S-GW function can include a request from the S-GW to a P-GW to allocate an IP address to a user device. Thus, south-facing functions in the core network, such as for the SGSN and S-GW functions that relate to the SGSN or the S-GW interacting with the RNC or the eNodeB respectively, can be retained in the LDC, whereas the north-facing functions in the core network, such as the SGSN and S-GW functions for interactions with the GGSN and the P-GW respectively, can be moved into the UDC. Additionally, functions in the LDC can be further split such that functions that require very high sensitivity support with respect to latencies such as functions that are related to the eNodeB or the RNC or a Small cell can be placed in a data

center (L1DC) that is placed further lower down in the network hierarchy (closer to users and/or devices), whereas functions that relate to the SGSN, the MME, and the S-GW can continue to be placed in the LDC. In such a scenario, we can call the LDC as the L2DC and the UDC as the L3DC as shown in Figure 8. Additional optimizations can include, for example, retaining the south-facing network functions in the radio access network (RAN) for the eNodeB, the RNC, and the Small cell VNFs, in the L1DC, whereas north-facing functions that relate to the interactions of the RNC with the SGSN, or the interactions of the eNodeB with the S-GW, can be moved up to the L2DC. An alternative example can include moving the southbound SGSN and S-GW functions into the L1DC, which can contain all of the functions for the RNC, eNodeB, or Small cells, whereas the northbound functions for the SGSN and S-GW can be retained in the L2DC. The upper data center L3DC can be placed in a cloud or within the operator's network, higher up in the network hierarchy.

B. Hierarchical Function Replication (HFR)

Functions can be replicated in the hierarchy as well. For example, highly mobile users can be served by an MME VNF in a DC that serves multiple access nodes (such as multiple eNodeBs) to handle mobility better based on a velocity threshold for handling mobile users, whereas static or quasi-static users can be served by an MME VNF in a lower DC closer to the access node (such as an eNodeB) where both replicated placements meet other latency/performance requirements for serving users. Billing functions can be replicated and distributed to optimize data routes. Additionally, eNodeBs can connect to more than one S-GW (hierarchical or collapsed placements), to provide different optimal user planes for data depending on the performance requirements for applications, such that different user devices could use different user planes, or the same user device could use different user planes for different applications if needed.

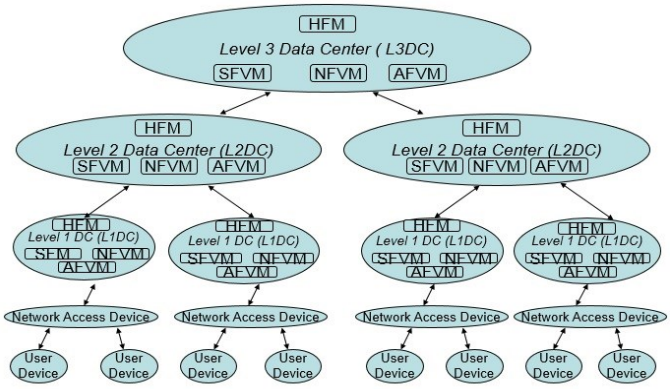

Figure 8: Hierarchical Partitioning for Network, Service or Application function VMs / Containers

In general, Distributed Hierarchy Function Managers (HFMs) can chain functions as needed, exchange info dynamically, and take decisions on placing functions optimally enabling Distributed Function Virtualization (DFV) while satisfying system constraints. Service Functions related to application proxies, or to support m2m services can be hosted closer to P-GW VMs. With the general data center hierarchy as shown in Figure 8, data center resources can be partitioned to support other applications function virtual machines (AFVMs) and service function virtual machines (SFVMs) containers as well in addition to supporting network function VMs (NFVMs).

SUMMARY AND CONCLUSIONS

With the availability of software appliances, one can host different network, service, and application functions at different levels in a data center hierarchy. The paper has suggested a methodology to address latency, cost, and policy constraints related to data center resources and VNF performance requirements to partition and place these functions across data centers optimally. Different variations on hierarchical placement of functions including hierarchical function collapsing, splitting, and replication were suggested. It is hoped that the proposed solutions in the paper will be useful to enhance capabilities in orchestration-related projects such as TOSCA, HEAT and openMANO in the OpenStack, OPNFV and nfvlabs communities [12], to enable network functions to be partitioned and mapped dynamically to available distributed hierarchical resources, thus providing good flexibility, configurability, and agility in emerging NFV systems.

REFERENCES

[1] "ETSI NFV Architecture Framework" http://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf

[2] G. Tseliou, F. Adelantado, C. Verikoukis, "Resources Negotiation for Network Virtualization in LTE-A Networks", ICC 2014, Sydney.

[3] S. Costanzo, D. Xenakis, N. Passas, L. Merakos, "OpeNB: A framework for Virtualizing Base Stations in LTE Networks", ICC 2014, Sydney.

[4] J. Chase, R. Kaewpuang, W. Yonggang, D. Niyato, "Joint Virtual Machine and Bandwidth Allocation in SDN and Cloud Computing Environments", ICC 2014, Sydney.

[5] Slavisa Aleksic, Igor Miladinovic, "Network Virtualization: Paving the way to carrier clouds", Proceedings of the 16th International Telecommunications Network Strategy and Planning Symposium (Networks 2014), September 2014, Funchal, Madeira Island, Portugal

[6] Clayman, S.; Maini, E.; Galis, A.; Manzalini, A.; Mazzocca, N., "The dynamic placement of virtual network functions," *Network Operations and Management Symposium (NOMS), 2014 IEEE* , vol., no., pp.1,9, 5-9 May 2014

[7] Davy, S.; Famaey, J.; Serrat-Fernandez, J.; Gorricho, J.L.; Miron, A.; Dramitinos, M.; Neves, P.M.; Latre, S.; Goshen, E., "Challenges to support edge-as-a-service," *Communications Magazine, IEEE* , vol.52, no.1, pp.132,139, January 2014

[8] Manzalini, A.; Minerva, R.; Callegati, F.; Cerroni, W.; Campi, A., "Clouds of virtual machines in edge networks," *Communications Magazine, IEEE* , vol.51, no.7, pp.63,70, July 2013

[9] NFVRG, https://datatracker.ietf.org/rg/nfvrg/charter/ .

[10] ILOG CPLEX, http://en.wikipedia.org/wiki/CPLEX

[11] Mininet, http://www.mininet.org

[12] Open source projects, http://www.openstack.org, http://www.opnfv.org, http://github.com/nfvlab