Resource Management in Softwarized Networks

Shihabur Rahman Chowdhury and Raouf Boutaba
David R. Cheriton School of Computer Science, University of Waterloo
{sr2chowdhury | rboutaba}@uwaterloo.ca

***Abstract*—Communication networks are undergoing a major transformation through *softwarization*, which is changing the way networks are designed, operated, and managed. The enhanced programmability enabled by softwarization creates unique opportunities for adapting network function and resources in support of applications and users with diverse requirements. To effectively leverage the flexibility provided by network softwarization and realize its full potential, it is of paramount importance to devise proper mechanisms for allocating resources to different applications and users and for monitoring their usage over time. The overarching goal of this dissertation is to advance state-of-the-art in how resources are allocated and monitored and build the foundation for effective resource management in softwarized networks. Specifically, we address four resource management challenges in all three key enablers of network softwarization. First, we challenge the current practice of realizing network services with monolithic software network functions and propose a microservice-based disaggregated architecture enabling finer-grained resource allocation and scaling. Then, we devise optimal solutions and scalable heuristics for establishing virtual networks with guaranteed bandwidth and guaranteed survivability against failures in multi-layer IP-over-Optical and single-layer IP substrate networks, respectively. Finally, we propose adaptive sampling mechanisms for balancing the overhead of softwarized network monitoring and the accuracy of the network view constructed from monitoring data.**

I. Introduction

Network Softwarization is an emerging paradigm where software controls the treatment of network flows, adds value to these flows by software processing, and orchestrates the on-demand creation of customized networks to meet the needs of customer applications [1]. Networking industry transformation through softwarization can be illustrated by the widespread adoption of the following technological developments.

Software-Defined Networking (SDN) proposes to decouple a network's control plane from the data plane, and implements the control plane as a logically centralized software *controller* running on one or more commodity servers [2].

Network Function Virtualization (NFV) proposes to decouple *Network Functions (NFs)* (*e.g.*, Network Address Translators (NATs), Firewalls, WAN Optimizers) from hardware *middleboxes*, and deploy the NFs as *Virtual Network Functions (VNFs)* on commodity servers [3].

Network Virtualization (NV) is a networking environment that allows coexistence of multiple *Virtual Networks (VNs)*, each tailored to support specific application or service, on a shared physical infrastructure [4].

Network softwarization is achieved by the amalgamation of SDN, NFV and NV, enabling on-demand service provisioning and better control over the network resources. Softwarization is replacing specialized hardware with commodity hardware with open architecture, in this way simplifying network infrastructure and diversifying the supply chain. Today, network softwarization is increasingly adopted by both large-scale online service providers and telecommunications and Internet service providers, and is considered a key enabler for 5G networks. Furthermore, it is creating new revenue streams by enabling service offerings which would have been otherwise very expensive to deploy and hard to manage with traditional networking technologies. However, to effectively leverage the flexibility provided by softwarization, it is of paramount importance to devise proper mechanisms for allocating resources to different applications and users and for monitoring their usage over time. Sub-optimal resource allocation can result in resource fragmentation and under-utilized infrastructure, which in turn results in loss of revenue.

978-1-6654-0601-7/22/$31.00

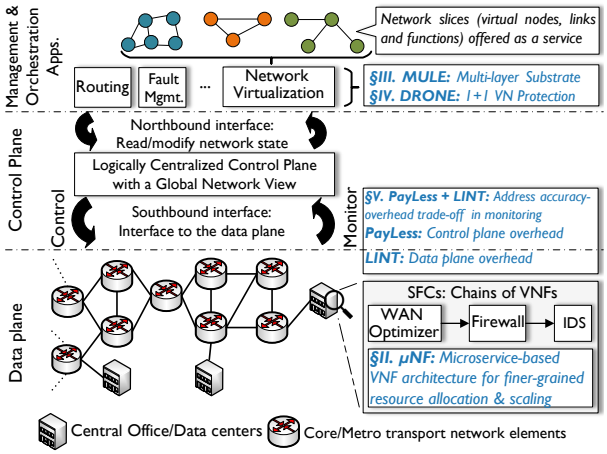


Fig. 1. Dissertation Contributions Across the Network Softwarization Stack

This dissertation advances the state-of-the-art in how resources are allocated and monitored, and builds a foundation for effective resource management in softwarized networks. We challenge some of the current practices as well as address the shortcomings in how resource allocation is performed and how the network infrastructure is monitored. As illustrated in Fig. 1, this dissertation addresses resource management issues encompassing SDN, NFV, and NV, substantially improving the effectiveness of resource allocation and striking a balance between accuracy and overhead in infrastructure monitoring. Our specific contributions are as follows:

- We challenge the current practice of realizing network services with monolithic VNFs and propose a microservice-based disaggregated VNF architecture for finer-grained resource allocation and scaling.
- We devise optimal solutions and scalable heuristics for establishing VNs with guaranteed bandwidth and guaranteed survivability against failures in multi-layer IP-over-Optical (Section III) and single-layer substrate networks (Section IV), respectively.
- We propose adaptive sampling mechanisms for balancing the overhead of softwarized network monitoring and the accuracy of the network view constructed from monitoring data (Section V).

II. A Disaggregated VNF Architecture

State-of-the-art NFV platforms (*e.g.*, OPNFV, OpenMANO) are *merely replacing monolithic hardware middleboxes with monolithic VNFs*. Clearly, this is a first logical step towards network softwarization. However, a fundamental problem with monolithic VNF implementations is that many packet processing tasks such as packet I/O, parsing and classification, and payload inspection are repeated across a wide range of NFs [5], [6] as illustrated in Fig. 2. This has several negative consequences. First, monolithic VNFs restrict how many packet processing tasks can be consolidated on the same hardware. Second, they impose coarse-grained resource allocation and scaling. Finally, when VNFs are chained to form *Service Function Chains (SFCs)*, executing these redundant functionalities results in unnecessary processing overhead (shown to exceed 25% for some SFCs [5]). This non-exhaustive list of issues stresses the need to rethink how VNFs can be developed and orchestrated for agile service creation and scaling. In this dissertation, we pose and address the following research question:

(Q1) *What is an appropriate software architecture for VNFs that will enable better function consolidation on the same hardware, and finer-grained resource allocation and scaling while maintaining the same level of performance as state-of-the-art approaches?*

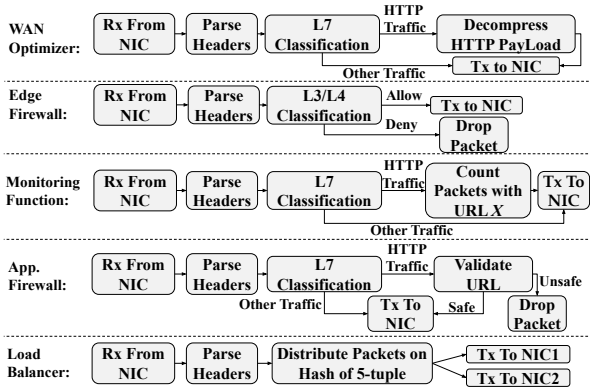


Fig. 2. Common Packet Processing Tasks Across NFs

A. Solution Overview

We take advantage of the commonality of packet processing tasks among VNFs for addressing the shortcomings of monolithic VNFs. We propose to decompose VNFs into independently deployable, loosely-coupled, lightweight, and reusable packet processors, that we call *MicroNFs* (μNFs for short) [7], [8]. VNFs or SFCs are then realized by composing a packet processing pipeline from these independently deployable μNFs. Such decomposition enables finer-grained resource allocation, independent scaling of μNFs thus increased flexibility, and independent development and maintenance of packet processing components.

A high level view of our system is presented in Fig. 3. It comprises the following components:

μNF Orchestrator responsible for making global decisions such as μNF placement across physical servers.

Orchestration agent, a per physical server local orchestration endpoint. A southbound API between the μNF orchestrator and orchestration agents facilitates their communication.

μNFs are the smallest deployable units in the system. μNFs perform a specific packet processing task and are independently deployable loosely-coupled entities. We keep the communication pattern between μNFs transparent from how they process the packets.

Rx and Tx Services collectively form a lightweight software data path for the μNFs.

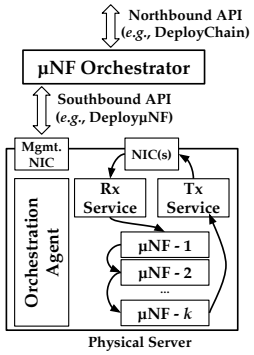
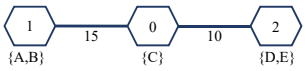


Fig. 3. μNF System Components

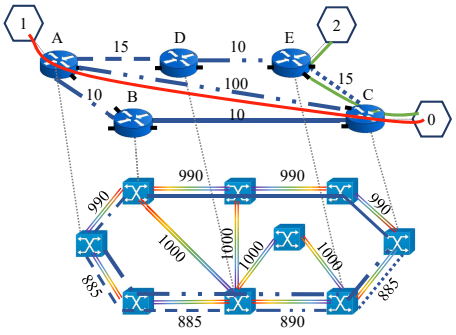
We have implemented the architecture components including the μNFs, communication primitives between μNF, and CPU sharing between μNFs to improve CPU utilization without sacrificing packet processing throughput. We also employed optimizations such as cache pre-fetching and μNF processing pipeline parallelization for improving packet processing throughput on multi-socket NUMA machines and for improving packet processing latency, respectively.

III. Multi-Layer Virtual Network Embedding

Softwarization is enabling the long-haul connectivity providers, *i.e.*, the transport network operators to offer full-fledged VNs to their customers [9] in lieu of traditional point-to-point connectivity services. This next generation of transport network, also known as *Transport SDN (T-SDN)*, leverages SDN technology to separate the control plane from the data plane (typically realized through a combination of



(a) Virtual Network Request



(b) VN Embedding on Multi-layer SN

Fig. 4. Multi-layer VN embedding example

packet and optical communications) for flexible management and better automation. Several novel and practically important resource allocation problems exist in this emerging area. For instance, multi-layer IP-over-Optical networks, which combine the high capacity of optical networks with the flexibility of packet switched networks, are becoming a popular choice of T-SDN deployment, However, *Virtual Network Embedding (VNE)*, a fundamental resource allocation problem in NV, has been mostly addressed for single-layer *Substrate Networks (SNs)*. Solving the VNE problem for multi-layer SN raises many unique challenges due to the topological flexibility offered by such networks [10]. Specifically, the IP network is dynamic, *i.e.*, new IP links can be established as needed by provisioning necessary capacity from the optical network. In this context, we pose the following research question:

***(Q2)** How can we leverage the topological flexibility of multi-layer IP-over-Optical T-SDN and: (i) strike a balance between obtaining a low cost VN embedding while minimizing the establishment of new IP links; (ii) simultaneously decide on the creation of new IP links and their embedding on the optical network?*

Several deployment choices exist for multi-layer T-SDN [11]. However, as a first step towards addressing VNE for multi-layer networks, we limit the scope of this work to the case of an IP-over-Optical Transport Network (OTN) [12].

A. Problem Statement

We represent the substrate OTN as an undirected graph $\hat{G} = (\hat{V}, \hat{E})$, where $\hat{V}$ and $\hat{E}$ are the set of OTN nodes and OTN links, respectively. The OTN links $(\hat{u}, \hat{v}) \in \hat{E}$ have two attributes: (i) $b_{\hat{u}\hat{v}}$: pre-provisioned bandwidth capacity; (ii) $C_{\hat{u}\hat{v}}$: the cost of allocating one unit of bandwidth from that OTN link. We represent the substrate IP network as an undirected graph $G' = (V', E')$, where each IP node $u' \in V'$ has $p_{u'}$ number of ports with homogeneous capacity $cap_{u'}$. Each IP node u' is connected to an OTN node $\tau(u')$. We use the notation (u', v', i) to represent the i-th IP link between u' and v', where $1 \leq i \leq p_{u'}$. An IP link (u', v', i) with bandwidth $b_{u'v'i}$ is provisioned by establishing an OTN path connecting its end points. Capacity of a new IP link (u', v', i) is set to $\min(cap_{u'}, cap_{v'})$. The cost of allocating one unit of bandwidth from an IP link $(u', v', i) \in E'$ is $C_{u',v',i}$. We represent a VN request as an undirected graph $\bar{G} = (\bar{V}, \bar{E})$, where $\bar{V}$ and $\bar{E}$ are the set of virtual nodes and virtual links, respectively. Each virtual link $(\bar{u}, \bar{v}) \in \bar{E}$ has bandwidth requirement $b_{\bar{u}\bar{v}}$. Each virtual node $\bar{u} \in \bar{V}$ has a location constraint set $\mathcal{L}(\bar{u}) \subset V'$ representing the set of IP nodes where $\bar{u}$ can be embedded.

Given a multi-layer SN composed of an IP network G' on an OTN $\hat{G}$, and a VN request $\bar{G}$:

- Map each virtual node $\bar{u} \in \bar{V}$ to an IP node $u' \in V'$ according to the virtual node's location constraint $\mathcal{L}(\bar{u})$.
- Map each virtual link $(\bar{u}, \bar{v}) \in \bar{E}$ to a path in the IP network. This path can contain a combination of existing IP links and newly created IP links.
- Map all newly created IP links to a path in the OTN.
- The total cost of provisioning resources for new IP links and for the virtual links should be minimized.

We give an illustrative example of the problem in Fig. 4. Here, we provision a new IP link AC (red link in Fig. 4(b)) by mapping it to OTN-layer (the bottom layer in Fig. 4(b)) in the process of establishing a VN.

B. Solution Overview

For the **MUL**ti-layer virtual network **E**mbedding (MULE) problem, we propose the following solutions [13], [14].

OPT-MULE: An Integer Linear Program (ILP) formulation for optimally solving MULE. Our ILP formulation jointly optimizes VN embedding, creation of new IP links and embedding of newly created IP links on the OTN layer. We also prove that optimally solving MULE is NP-hard. To the best of our knowledge, this is the first optimal solution to the VNE problem for multi-layer IP-over-OTN networks.

FAST-MULE: Given the NP-Hard nature of optimally solving the multi-layer VNE problem, we propose, FAST-MULE, a heuristic for solving larger problem instances. We are faced with two challenges while designing FAST-MULE: performing (i) joint VN embedding across the IP and the OTN layers; and (ii) joint virtual node and link mapping. We address the first challenge by transforming the multi-layer SN into a single-layer graph, in this way jointly computing embedding across the layers. For addressing the second challenge, we extract star subgraphs from the VN and jointly embed the nodes and links of the subgraph by transforming the embedding problem to an instance of the max-flow problem.

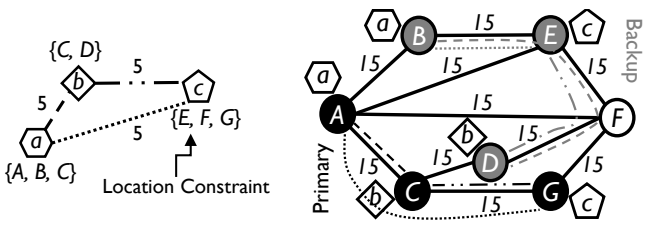
IV. Dedicated Protection for Survivable VNE

Another challenge in T-SDN virtualization concerns guaranteeing VN survivability against failures. T-SDN customer VNs typically carry high volumes of traffic at high speed, and usually have Service Level Agreements (SLAs) with the infrastructure provider for recovery from substrate failures within tens of milliseconds [15]. One way to satisfy such tight SLA is that the infrastructure provider provisions dedicated backup resources for the entire VN (*i.e.*, for each virtual node and link in a VN), also known as 1+1 protection scheme. Backup of the entire VN topology can be later used for immediate recovery from a substrate failure [16]. However, such fast recovery with dedicated backup is expensive for the infrastructure provider. The research literature lacks a systematic approach to solving the problem with optimal resource footprint. Furthermore, relevant literature [17] shows that sequentially embedding the primary and backup can lead to failure in embedding even though a feasible embedding exists, which adds another dimension when designing a heuristic. In this context, we pose the following research question:

***(Q3)** How can we simultaneously compute the primary and the backup embedding of a VN for dedicated protection while jointly determining virtual node and virtual link embedding and incurring the minimum resource footprint in the substrate network?*

A. Problem Statement

We represent the SN as an undirected graph, $G = (V, E)$, where V and E are the set of substrate nodes and links, respectively. Each substrate link $(u, v) \in E$ has the following



(a) Virtual Network Request (b) Embedding on SN by DRONE

Fig. 5. Example VN embedding with DRONE

attributes: (i) b_{uv} : bandwidth capacity of the link (u, v), and (ii) C_{uv} : cost of allocating unit bandwidth on (u, v) for provisioning a virtual link. We represent a VN as an undirected graph $\bar{G} = (\bar{V}, \bar{E})$, where $\bar{V}$ and $\bar{E}$ are the set of virtual nodes and virtual links, respectively. Each virtual link $(\bar{u}, \bar{v}) \in \bar{E}$ has bandwidth requirement $b_{\bar{u}\bar{v}}$. The VN has a location constraint set, $\mathcal{L} = \{L(\bar{u}) | L(\bar{u}) \subseteq V, \forall \bar{u} \in \bar{V}\}$, such that a virtual node $\bar{u} \in \bar{V}$ can only be provisioned on a substrate node $u \in L(\bar{u})$.

Given a SN $G = (V, E)$, VN request $\bar{G} = (\bar{V}, \bar{E})$, and a set of location constraints $\mathcal{L}$, embed $\bar{G}$ on G such that:

- Each virtual node $\bar{u} \in \bar{G}$ has a primary and a backup embedding in the SN, satisfying the location constraint.
- For each virtual node $\bar{u} \in \bar{G}$, the substrate nodes used for the primary embedding are disjoint from the substrate nodes used for the backup embedding.
- Each virtual link $(\bar{u}, \bar{v}) \in \bar{E}$ has a primary and a backup embedding in the SN. A primary or backup embedding of a virtual link on the SN corresponds to a single path in the SN having at least $b_{\bar{u}\bar{v}}$ available bandwidth. The substrate paths corresponding to the primary and backup embedding of a virtual link $(\bar{u}, \bar{v}) \in \bar{E}$ are represented by $P_{\bar{u}\bar{v}}$ and $P'_{\bar{u}\bar{v}}$, respectively.
- Backup embedding of a virtual link is disjoint from the set of substrate paths used for primary embedding of the virtual links. The same disjointedness principle applies for the primary embedding.
- The total cost of provisioning bandwidth in the SN is minimum according to the following cost function:

$$\sum_{\forall (\bar{u},\bar{v}) \in \bar{E}} \sum_{\forall (u,v) \in P_{\bar{u}\bar{v}} \cup P'_{\bar{u}\bar{v}}} C_{uv} \times b_{\bar{u}\bar{v}} \qquad (1)$$

Fig. 5 shows an illustrative example with dark nodes and lines representing the primary, and grey nodes and lines representing the backup embedding of a VN on an SN.

B. Solution Overview

We propose **D**edicated P**ro**tection for Virtual **N**etwork **E**mbedding (DRONE), a suite of solutions for the 1 + 1-**Pro**tected **Vi**rtual **N**etwork **E**mbedding (1 + 1 – ProViNE) problem [18], [19]. We focus on single node failure scenario since it is the most probable case [20], [21]. Specifically, we make the following contributions.

OPT-DRONE: An ILP formulation for optimally solving 1 + 1 – ProViNE, improving on the quadratic formulation from previous work [17]. We formulate 1 + 1 – ProViNE as simultaneously embedding two copies of the same VN disjointly on the SN. To accomplish this goal, we replicate the input VN, $\bar{G}$ to obtain a shadow VN, $\tilde{G} = (\tilde{V}, \tilde{E})$. Our transformed input now contains the graph $\hat{G} = (\hat{V}, \hat{E})$, *s.t.* $\hat{G} = \bar{G} \cup \tilde{G}$. We now embed $\hat{G}$ on G *s.t.* any node $u \in \bar{V}$ and any node $\tilde{u} \in \tilde{V}$ are not provisioned on the same substrate node. Similar constraints apply on the virtual links as well.

FAST-DRONE: OPT-DRONE is NP-hard. Therefore, we also devise FAST-DRONE, a heuristic for finding solutions within a reasonable time frame. For the ease of designing an effective heuristic, we reformulate 1 + 1 – ProViNE as a variant of graph partitioning problem as follows. Given an SN $G = (V, E)$, a VN request $\bar{G} = (\bar{V}, \bar{E})$, and a set of location constraints $\mathcal{L}$, solving 1 + 1 – ProViNE entails partitioning G into two disjoint partitions $\mathcal{P}$ and $\mathcal{Q}$ such that:

- $\forall \bar{u} \in \bar{V}$, $\mathcal{P}$ has at least one element from each $L(\bar{u})$.
- $\forall \bar{u} \in \bar{V}$, $\mathcal{Q}$ has at least one element from each $L(\bar{u})$.
- The sub-graph induced by the elements of each set $L(\bar{u})$ in $\mathcal{P}$ (and $\mathcal{Q}$) is connected.
- The sum of costs of embedding $\bar{G}$ on $\mathcal{P}$ and $\mathcal{Q}$ is minimum according to the given cost function (1).

We leverage this new problem formulation and devise a heuristic that starts with a seed mapping set containing the primary and backup mappings of one randomly chosen virtual node. Then we iteratively grow this seed mapping to compute the primary and backup mappings of the remaining virtual nodes and partition the SN in this process. Once the SN is partitioned, we embed the virtual links in these partitions separately by using the Constrained Shortest Path First algorithm.

V. Adaptive Monitoring of Softwarized Networks

The trade-off between accuracy and overhead has been a longstanding issue in network monitoring. This issue is further aggravated by the flexibility brought by network softwarization. Softwarization is enabling the network operators to program a wide-range of network resources ranging from the software switches, the software network functions, and even the hardware data plane for collecting fine-grained and near real-time network monitoring data. A direct consequence of this advantage is the increased monitoring overhead in both the control and the data planes [22], [23]. In this context, we address the following research question:

***(Q4)** How can we construct an accurate and timely view of the network without incurring significant control plane and data plane overhead for network monitoring?*

A. Solution Overview

Our approach to addressing the accuracy – overhead trade-off in softwarized network monitoring is to identify *less interesting* observations while collecting monitoring data and adapt the monitoring accordingly. Our objective is to reduce both the control and data plane overhead without negatively impacting the quality of collected monitoring data. To this end, we propose the following adaptive monitoring mechanisms for the control and data plane, respectively.

1) PayLess [24] (control plane): SDN control plane constructs a global network view by querying the data plane for traffic statistics, which consumes control plane bandwidth and incurs message processing overhead at the controller [22]. PayLess is a variable frequency SDN monitoring algorithm

that adapts monitoring frequency according to traffic intensity. An SDN controller running PayLess assigns a statistics collection timeout τ to each new flow. If the flow is still active after τ time units, a statistics collection event triggers and the controller queries the corresponding switch for reading that flow's counters (*e.g.*, packet and/or byte count). Depending on the change in the counter value since the last measurement, the controller adjusts the timeout τ of that flow as follows:

- If the change is small, *i.e.*, below a threshold Δ_1, τ for that flow is multiplied by a factor α (capped at $\mathcal{T}_{max}$).
- If the change is above a threshold Δ_2, τ of that flow is reduced by a factor β (capped at $\mathcal{T}_{min}$).

The rationale behind this timeout adjustment is that flows that significantly changed in volume (bytes or packets) have higher chances of contributing to triggering interesting network events (*e.g.*, become heavy-hitters). Therefore, once we detect such change, we monitor that flow more frequently. For a similar reason, we reduce the polling frequency of the steady flows since they are not conveying much information.

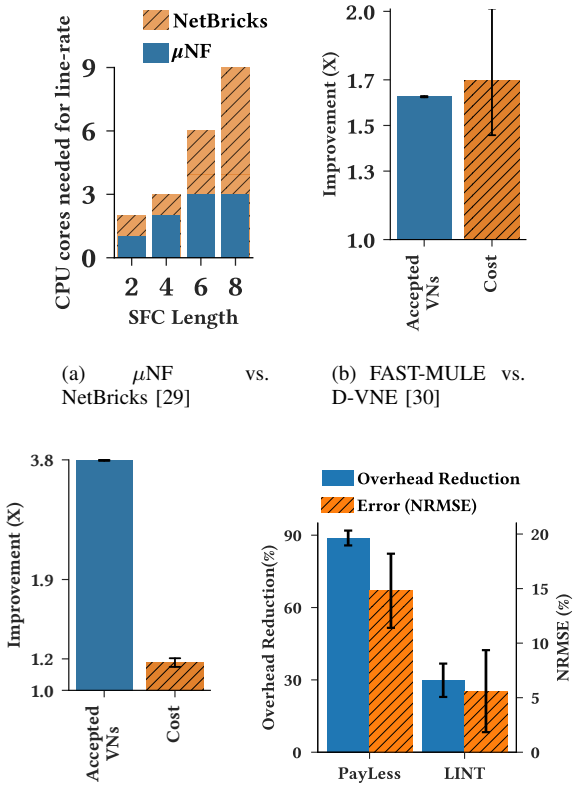
2) LINT [25] (data plane): In-band Network Telemetry (INT) [26] is an emerging standard for enabling network devices to embed device internal state such as packet processing latency and queue depth into each passing packet, consequently, facilitating a real-time and microscopic view into network traffic. The microscopic telemetry capabilities enabled by INT come at the expense of increased data plane overhead [23] stemming from piggybacking telemetry data on live network traffic. Our empirical study using real network traces revealed that INT data plane overhead can result in 20% - 30% reduction in network goodput [25]. Therefore, we proposed LINT (short for **L**ightweight **INT**), which aims at reducing INT data plane overhead while reaping its benefits as much as possible. We design LINT to be implementable within the constraints of commodity programmable data plane devices [27] and capable of working without any global coordination and intervention from a control plane.

We leverage model-driven data acquisition techniques from sensor networking literature [28] for identifying and filtering *less interesting* telemetry data directly in the data plane without negatively impacting the quality of collected telemetry data. A network device running LINT tries to estimate the amount of error that can be introduced if the requested telemetry data items are not piggybacked on the current packet. For estimating this error, the device uses a predictor function (Exponentially Weighted Moving Average (EWMA) in our implementation) for each telemetry data item of interest. The predictor function for a telemetry data item d is used for computing the following:

- d_D^{next}: the predictor function applied on all past observations of d in this device.
- d_C^{next}: the predictor function applied on the observations of d reported so far.

When the device predicts that these two quantities may diverge beyond an acceptable threshold δ, *i.e.*, $|d_D^{next} - d_C^{next}| > \delta \times d_D^{next}$, it reports the current observation. Otherwise, the device skips reporting the current observation, in this way, adapting telemetry data reporting based on estimated error.

VI. Summary of Results



(a) μNF vs. NetBricks [29]

(b) FAST-MULE vs. D-VNE [30]

(c) FAST-DRONE vs. PAR [17]

(d) Effectiveness of PayLess & LINT

Fig. 6. Highlights of thesis contributions: (a) effectiveness of μNF in enabling finer-grained resource allocation compared to NetBricks [29]; (b) VN acceptance and embedding cost comparison between FAST-MULE and D-VNE [30]; (c) VN acceptance and embedding cost comparison between FAST-DRONE and PAR [17]; and (d) effectiveness of PayLess and LINT in reducing control and data plane overhead, respectively.

We have evaluated our solutions using a variety of methods including, testbed experiments, network emulation and simulations. We leveraged publicly available real network topologies and real traffic traces when possible. In many cases, we also implemented state-of-the-art solutions and compared our contributions against them. In this section and in Fig. 6, we summarize and present some key results of our contributions.

We first demonstrate the finer-grained resource allocation capability of μNF compared to state-of-the-art run-to-completion monolithic SFC deployment system NetBricks [29] in Fig. 6(a). We deploy SFCs of varying lengths using both μNF and NetBricks on a testbed and measure the number of CPU cores they require for sustaining line-rate packet processing throughput (for the smallest packet size). We found μNF to be always using the same or lesser number of CPU cores compared to NetBricks. Furthermore, the CPU core consumption gap between μNF and NetBricks widens as the SFCs become longer.

Following the effectiveness of fine-grained resource allocation, we demonstrate the performance of the resource allocation heuristics proposed for MULE and 1 + 1 – ProViNE in Fig. 6(b) and Fig. 6(c), respectively. In both cases, we exten-

sively evaluate the solutions using realistic network topologies and compare the results against state-of-the-art heuristics. In case of MULE, our proposed heuristic, FAST-MULE, accepts $\approx$1.6$\times$ more VNs and incurs $\approx$1.7$\times$ less resource provisioning cost on average compared to state-of-the-art heuristic [30]. FAST-MULE also finds the optimal solution for a special class of VN, *i.e.*, star topology with uniform bandwidth requirement. FAST-DRONE outperforms the corresponding state-of-the-art heuristic [17] by accepting $\approx$3.8$\times$ more VNs and incurring $\approx$1.17$\times$ less resource provisioning cost on average.

Finally, Fig. 6(d) shows the effectiveness of PayLess and LINT in reducing control and data plane overhead for softwarized network monitoring, respectively. We evaluate our contributions using network emulation and simulation using both synthetic and real traffic traces. Our key results are: (i) PayLess can reduce control plane messaging overhead by more than 80% and incurs $\approx$20% normalized root means squared error (NRMSE) of link utilization measurement on average compared to periodically polling the switches in 250 ms intervals; and (ii) on average, LINT reduces the data plane overhead of INT by $\approx$30% while incurring $\approx$5% NRMSE for switch processing latency measurement.

VII. Conclusion

Telecommunications and data center network operators are increasingly adopting network softwarization for diversifying their supply chain, enabling on-demand service provisioning, achieving better control over the network resources, and accelerating the time-to-market for new services. However, reaping the full benefits of softwarization requires resource management mechanisms that can take full advantage of the flexibility brought forth by softwarization. In this context, this dissertation challenges some of the current resource management practices and addresses the shortcomings in how resource allocation is performed and how softwarized networks are monitored. Specifically, it addressed four resource management challenges in three key enablers of network softwarization, namely SDN, NFV, and network virtualization, paving the way for effective resource management in next generation communication networks. We have demonstrated the effectiveness and superiority of our proposed solutions through extensive testbed evaluation, network emulation and simulations using realistic scenarios, and comparing to state-of-the-art solutions when possible. We believe that the dissertation research has contributed to solving some fundamental problems underlying network softwarization, which we hope will facilitate its further adoption along with the use of AI for achieving closed loop automated network management.

VIII. Final Remarks

This thesis is available at http://hdl.handle.net/10012/16824. Research conducted as part of this dissertation has been published in [6]–[8], [13], [14], [18], [19], [24], [25], [31] and has resulted into two patents granted [32], [33] (the full list of publications and patents along with bibliometric data is available at https://scholar.google.com/citations?user=r_AbXNsAAAAJ). Part of the work that appeared in this dissertation has received conference best paper awards at IEEE/ACM/IFIP CNSM 2017 [13] and IEEE NetSoft 2019 [7]. Furthermore, this dissertation has been recognized with the 2021 Alumni Gold Medal from the University of Waterloo.

References

[1] H. Freeman and R. Boutaba. Networking industry transformation through softwarization [the president's page]. *IEEE COMMAG*, 54(8):4–6, August 2016.

[2] M. Casado et al. Ethane: Taking control of the enterprise. In *ACM SIGCOMM*, pp. 1–12, 2007.

[3] Network Functions Virtualisation – Introductory White Paper, Oct 2012.

[4] M. Chowdhury and R. Boutaba. A survey of network virtualization. *Elsevier Computer Networks*, 54(5):862–876, April 2010.

[5] V. Sekar et al. Design and implementation of a consolidated middlebox architecture. In *USENIX NSDI*, pp. 323–336, 2012.

[6] S. R. Chowdhury et al. Re-architecting nfv ecosystem with microservices: State of the art and research challenges. *IEEE Network*, 33(3):168–176, 2019.

[7] S. R. Chowdhury et al. μNF: A Disaggregated Packet Processing Architecture. In *IEEE NetSoft*, 2019.

[8] S. R. Chowdhury et al. A Disaggregated Packet Processing Architecture for Network Function Virtualization. *IEEE JSAC*, 38(6):1075–1088, 2020.

[9] SDN Architecture for Transport Networks. White paper, March 2016.

[10] X. Jin et al. Optimizing bulk transfers with software-defined optical wan. In *ACM SIGCOMM*, pp. 87–100, 2016.

[11] F. Rambach et al. A multilayer cost model for metro/core networks. *IEEE/OSA JOCN*, 5(3):210–225.

[12] ITU-t recommendation g.709/y.1331: Interfaces for the optical transport network. Technical Report, 2016.

[13] S. R. Chowdhury et al. Mule: Multi-layer virtual network embedding. In *Proc. of IEEE/ACM/IFIP CNSM*, pp. 1 – 9, 2017.

[14] S. R. Chowdhury et al. Multi-layer virtual network embedding. *IEEE TNSM*, 15(3):1132–1145, 2018.

[15] S. Ramamurthy and B. Mukherjee. Survivable WDM mesh networks. Part I-protection. In *IEEE INFOCOM*, pp. 744–751, 1999.

[16] W. Wang et al. First demonstration of virtual transport network services with multi-layer protection schemes over flexi-grid optical networks. *IEEE Comm. Lett.*, 20(2):260–263, 2016.

[17] Z. Ye et al. Survivable virtual infrastructure mapping with dedicated protection in transport software-defined networks [invited]. *IEEE/OSA JOCN*, 7(2):183–189, 2015.

[18] S. R. Chowdhury et al. Protecting virtual networks with drone. In *IEEE/IFIP NOMS*, 2016.

[19] S. R. Chowdhury et al. Dedicated protection for survivable virtual network embedding. *IEEE TNSM*, 13(4):913–926, 2020.

[20] P. Gill et al. Understanding network failures in data centers: measurement, analysis, and implications. *ACM CCR*, 41(4):350–361, 2011.

[21] A. Markopoulou et al. Characterization of failures in an operational ip backbone network. *IEEE/ACM ToN*, 16(4):749–762, 2008.

[22] A. Yassine et al. Software defined network traffic measurement: Current trends and challenges. *IEEE Instr. & Measurement Mag.*, 18(2):42–50, April 2015.

[23] R. Ben Basat et al. Pint: Probabilistic in-band network telemetry. In *ACM SIGCOMM*, pp. 662–680, 2020.

[24] S. R. Chowdhury et al. PayLess: A low cost network monitoring framework for software defined networks. In *IEEE/IFIP NOMS*, 2014.

[25] S. R. Chowdhury et al. LINT: Accuracy-adaptive and Lightweight In-band Network Telemetry. In *IFIP/IEEE IM*, 2021.

[26] T. P. A. W. Group. In-band Network Telemetry (INT) data plane specification, June 2020.

[27] P. Bosshart et al. Forwarding metamorphosis: Fast programmable match-action processing in hardware for sdn. In *ACM SIGCOMM*, 2013.

[28] D. Goldsmith and J. Brusey. The spanish inquisition protocol—model based transmission reduction for wireless sensor networks. In *IEEE SENSORS*, pp. 2043–2048, 2010.

[29] A. Panda et al. Netbricks: Taking the V out of NFV. In *OSDI'16*.

[30] J. Zhang et al. Dynamic virtual network embedding over multilayer optical networks. *IEEE/OSA JOCN*, 7(9):918–927, September 2015.

[31] S. R. Chowdhury et al. ReViNE: Reallocation of virtual network embedding to eliminate substrate bottlenecks. In *IFIP/IEEE IM*, 2017.

[32] R. Ahmed et al. Dedicated protection for virtual network embedding, 2019. U.S. Patent No. 10,313,195. Issued June 04, 2019.

[33] S. R. Chowdhury et al. Multi-layer virtual network embedding, 2021. U.S. Patent No. 10,951,317. Issued March 16, 2021.