Machine Learning Methods for Slice Admission in 5G Networks

Machine Learning Methods for Slice Admission in 5G Networks

Muhammad Rehan Raza[1], Carlos Natalino[2], Lena Wosinska[2], Paolo Monti[2]
[1]KTH Royal Institute of Technology, Electrum 229, SE-164 40 Kista, Sweden
[2]Chalmers University of Technology, Chalmersplatsen 4, 41296, Gothenburg
e-mail: mrraza@kth.se; {carlos.natalino, wosinska, mpaolo}@chalmers.se

Abstract: *The paper discusses how the slice admission problem can be aided by machine learning strategies. Results show that both supervised and reinforcement learning might lead to profit maximization while containing losses due to performance degradation.*
Keywords: *Data analytics for network control and management in optical core/data center networks; Optical core/metro/data-center network architecture, design, virtualization, slice, control and management*

I. INTRODUCTION

In the 5G paradigm, infrastructure providers (InPs) will have to support a variety of services over a single network infrastructure to improve resource use efficiency and to reduce cost [1]. Software defined networking (SDN) and network function virtualization (NFV) are two promising technologies to enable such a vision [2]. Thanks to SDN and NFV, an InP can share its resources among different tenants, i.e., a concept referred to as *slicing*. Following this intuition, resources are allocated to tenants according to their specific requirements (e.g., latency, capacity, reliability). In the presence of temporal variations of such requirements, an InP can provision slices that can be scaled up/down during their service time [1]. In this scenario, the profit of an InP can be maximized by: *(i)* accepting as many slice requests as possible (i.e., to increase revenue), and *(ii)* matching the variation of the slice requirements as closely as possible (i.e., to contain penalties proportional to performance *degradation*, if a slice cannot be scaled up due to resource contention) [3]. In this respect, it becomes crucial to have an intelligent slice admission policy that accepts only slices which are not likely to create performance degradation in the network. Machine learning (ML) might help in this process by *(i)* understanding when/where resource bottlenecks might appear in the infrastructure, and *(ii)* deciding which slice to accept in order to maximize the profit of an InP. The paper discusses two use cases where ML-based slice admission policies are used to maximize the number of accepted slices (i.e., revenue) while minimizing their degradation levels (i.e., penalty). Results show that ML can significantly increase the profit of InPs as compared to conventional (i.e., not ML-based) slice admission heuristics.

II. NETWORK SLICING AND MACHINE LEARNING

ML algorithms can be categorized in *(i)* supervised learning (SL), *(ii)* unsupervised learning (UL), and *(iii)* reinforcement learning (RL) [4]. This paper focuses on two types of ML algorithms applied to the slice admission problem (i.e., SL and RL). The SL-based solution leverages on big data analytics (BDA) to process network historical data and to gather insight on how the requirement of a slice may vary over time. This information becomes crucial to understand if (upon being accepted) a slice may create problems to the slices already provisioned in the infrastructure, or may experience degradation itself. On the other hand, the RL-based solution relies on an agent able to understand under which conditions resource bottlenecks may appear (over time) in the infrastructure. Based on this information the agent then accepts only those slices likely to experience/create less or no degradation at all.

Looking at the literature, some BDA-based network slicing policies have been recently proposed. The work in [5] presents a strategy by which slices can be proactively scaled up/down according to BDA-based resource prediction. A similar approach is experimentally demonstrated by the work in [6]. The authors in [7] aim at minimizing degradation and show how BDA predictions can help while scaling up/down slices with different priorities. On the other hand, there is not much work available in the literature related to RL-based network slicing methods. The authors in [8] are among the first to assess the benefits of RL while scaling up/down slices according to traffic patterns of mobile users. The list of works mentioned so far focuses on the slice scaling problem. Applying DBA or RL to the slice admission process has only been recently investigated in [3] and [9]. The remainder of the paper summarizes the intuition and the benefits of the slice admission policies proposed in these works.

III. SLICE ADMISSION USING BIG DATA ANALYTICS: USE CASE DEFINITION AND PERFORMANCE EVALUATION

Figure 1(a) presents the system architecture for the use case under exam [3]. The *optical transport network*, managed by a transport controller, is based on dense wavelength division multiplexing (DWDM) technologies. The *radio domain,* managed by a radio controller, provides mobile broadband services using cells deployed according to the centralized-radio access network (C-RAN) concept. The *cloud domain*, managed by a cloud controller, provides

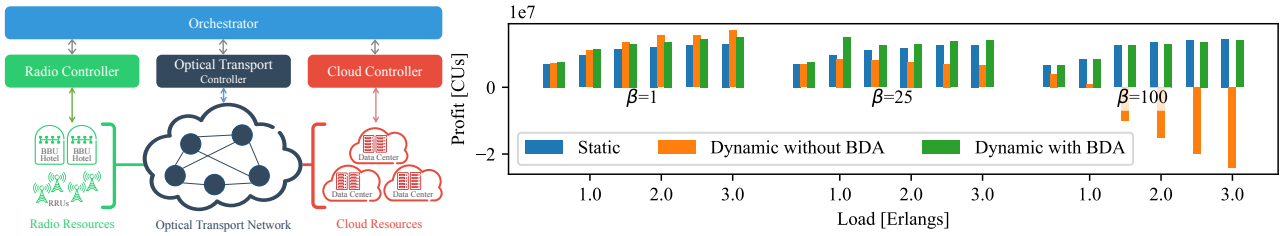

Fig. 1. (a) System architecture, (b) profit results with different values of β.

compute/storage services via datacenters interconnected in the optical domain. On top of these three domains, an *orchestrator* performs cross-domain management of infrastructure resources and creates/scales slices assigned to different tenants. The orchestrator is also enriched with BDA functionalities for predicting the temporal variation of the resource requirements of each slice. The use case under exam assumes two types of tenants (i.e., *mobile* and *cloud* providers). Upon reception of a slice request, the orchestrator checks the resource availability in the infrastructure and, based on the slice requirements, decides whether or not to accept the request. Each accepted slice generates revenue proportional to the slice duration. During their service time, slices are scaled up/down to match the temporal variations of their requirements. When slices cannot be scaled up, a penalty is paid to the tenant according to the experienced service degradation, which in turn has an impact on the resulting profit of the InP. A factor β is defined as the ratio between degradation penalty and the loss of revenue (i.e., due to slice rejection). With β=1 the degradation penalty is equivalent to the loss of revenue, while with β>1 the degradation penalty is higher than the loss of revenue. For more details on the use case and on the cost function applied to compute the profit, we refer to [3].

In order to limit degradation, BDA predictions can be used during the slice admission process, as follows. For each incoming request, the orchestrator retrieves a prediction for the requirements of the incoming slice and of each slice currently provisioned. Additionally, the orchestrator checks the current utilization of infrastructure resources. If accepting an incoming slice would result in a possible degradation of either the incoming or any of the slices currently provisioned, the orchestrator rejects the slice request. This guarantees that there will not be any degradation at all if the BDA predictions are 100% accurate. However, in case of inaccuracies, one or more slices may still experience degradation. Nonetheless, the proposed policy rejects altogether those slice requests which are expected to incur and/or create resource contentions. In this way, the penalty due to degradation is minimized, with beneficial effects on the profit of an InP. More details on this slice admission policy are available in [3].

The performance of the BDA-based slice admission policy described above is evaluated by applying it to the *dynamic slicing* approach presented in [1]. We consider a 12-node network topology [8] covering two geo-types (i.e., business and residential) comprising 6 nodes each. Slices can be of two types (i.e., mobile or cloud). The resource requirement profiles of each geo-type along with more details on the simulation settings are provided in [3]. The performance of dynamic slicing with BDA is compared against two benchmarks, *(i) static slicing*, where an InP always assigns peak amount of resources to a slice, and *(ii) dynamic slicing without BDA*, where slices are scaled up/down dynamically but the BDA is not used in the slice admission phase. Figure 1(b) compares the InP profit (measured in cost units [CUs]) for three different β values. When β=1 [CU], dynamic slicing without BDA leads to the highest profit as the degradation is not highly penalized. When β=25 [CUs], the profit for dynamic slicing without BDA drops significantly at high loads due to the high degradation penalty. However, dynamic slicing with BDA generates 49% more profit under high load condition. When β=100 [CUs] (e.g., for extreme applications where degradation is not acceptable at all), static slicing must be used. More detailed performance analysis of this BDA-based slice admission policy is available in [3].

IV. Slice Admission Using Reinforcement Learning: Use Case Description and Performance Evaluation

In the second use case examined in the paper, the control plane is the same as the one described in Sec. III (i.e., an orchestrator manages resources administered by one radio, transport, and cloud controller). However, the data plane comprises a flexible RAN architecture [9] where baseband processing functions (BPF) are virtualized (Fig. 2(a)). More specifically, a BPF is interconnected to a virtualized packet processor (vPP) function, which carries the data to/from the packet gateway (PGW). BPFs run on special purpose processors at the central office (CO) located close to remote radio units (RRUs) to meet their short latency requirements. vPPs and PGWs functions run over general-purpose processors (GPPs) and can be virtualized at the COs or at the remote data center (RDC) sites, depending on the service latency constraints. In the latter case, they also need connectivity resources over the optical backhaul (OBH) network. The service latency constraint also governs whether service specific VNFs (i.e., generic APP) are placed at the COs or at the RDC sites. Two types of services are considered in the use case, i.e., high priority (HP) and low priority (LP). A HP service (i.e., with strict latency constraints) requires a slice of GPPs (i.e., to run vPPs, PGWs, and APP functions) placed at the CO. However, a LP service (i.e., with non-strict latency constraints) may use GPPs placed at the CO and/or at RDC sites. Since GPPs resources at the CO are limited compared to the ones at RDC sites, using them is more costly. As a result, the tenants need to pay more for provisioning the HP services. Consequently, HP services generate

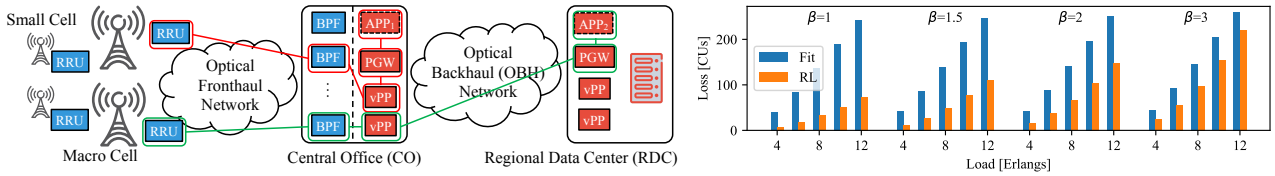

Fig. 2. (a) Flexible RAN architecture and (b) Penalty results for different values of β.

higher revenue for InP than LP services. A slice is scaled up/down in order to match the temporal variations in the number of required GPPs and (if allocated) OBH resources. A revenue loss is experienced if a slice request is rejected, and a penalty is paid if (at any point of time) the required number of GPP and/or OBH resources cannot be provisioned to a slice.

In order to limit the degradation, an RL method can be used during the slice admission process, as follows. When a slice is accepted, the orchestrator is responsible for provisioning it and for scaling it when necessary. Scaling is performed periodically for all the slices currently in the network, in order to match the temporal variations of their requirements. At the end of each scaling cycle, the degradation experienced by each slice, if any, is computed. Upon a slice teardown, a reward is computed, which is proportional to the total revenue obtained by operating the slice during its service time, minus the degradation penalty experienced, if any. The value of the reward is then fed to a RL agent that, in turn, learns the relationship between an action taken (slice acceptance/rejection) and the resulting value of the overall profit. The RL agent is modeled as a stochastic policy network (PN) which uses an artificial neural network (ANN) to represent its policy. The ANN receives as input an array describing how GPP/OBH resources are currently used and the requirements of the slice request. The ANN has two outputs (i.e., the probability of accepting or rejecting the slice request) which dictate the action taken by the orchestrator. The work in [9] describes in detail the reward function and ANN training procedure. The overall objective of the admission policy is loss minimization, where the loss has two components: *(i)* loss of revenue derived by rejecting slice requests, and *(ii)* the loss due to degradation.

The performance of the RL-based slice admission policy is assessed using the 12-node topology described in Sec. III. Five COs and two RDCs are placed at high degree nodes. HP services generate revenue 5 times higher than LP services, but they also may incur 5 times higher degradation penalties. Regardless of the priority, the degradation of each service leads to a penalty proportional to β. The ratio between HP and LP services is assumed to be 50%. The mean holding time of a slice is 1 day. More details on the simulation settings and on the requirements of each slice are specified in [8]. The scaling is done using a heuristic algorithm, which scales all the HP services before the LP ones. The performance is benchmarked against a Fit policy, which accepts a slice request if and only if all the currently required resources are available. Figure 2(b) compares the loss for four different values of β. The results show that with β=1 or β=1.5, RL leads to lower loss (80% in low loads and 64% in high loads) compared to Fit because the degradation of an accepted slice results in a very low penalty. On the other hand, when the value of β increases the gain of RL over Fit decreases. At low load and β=3, there is a 39% improvement, which becomes 14% at high load conditions. In this case, the RL agent understands that the degradation of an accepted service causes high penalty. As a result, it starts rejecting some of the slice requests to limit the value of the total loss.

V. Conclusions

This paper discusses how ML-based algorithms can improve the performance of slice admission control strategies in 5G networks. In the first use case, a BDA-based algorithm is used to predict the time-varying resource requirement of incoming slice requests and of the slices already in service. In the second use case, a RL algorithm understands the relationship between an action taken (slice acceptance/rejection) and the resulting value of the overall profit. Results show that with the help of ML-based methods, the losses of an InP can be substantially reduced.

Acknowledgment

This work was supported in part by the Kista 5G Transport Lab (K5) project funded by VINNOVA and Ericsson.

References

[1] M. R. Raza et al., "Dynamic slicing approach for multi-tenant 5G transport networks," JOCN, vol. 10, no. 1, Jan. 2018.
[2] J. Ordonez-Lucena, et al. "Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges." IEEE Commun. Mag., vol. 55, no. 5, 2017.
[3] M. R. Raza et al., "A slice admission policy based on big data analytics for multi-tenant 5G networks," JLT, Jan. 2019.
[4] F. Musumeci et al., "An overview on application of machine learning techniques in optical networks," in IEEE Commun. Surveys Tuts, 2018.
[5] L. Gifre et al., "Big data analytics in support of virtual network topology adaptability," OFC, Mar. 2016, paper W3F.6.
[6] L. Velasco et al., "An architecture to support autonomic slice networking," JLT, vol. 36, no. 1, Jan. 2018.
[7] M. R. Raza et al., "Priority-aware service orchestration using big data analytics for dynamic slicing in 5G transport networks," ECOC, Sep. 2017.
[8] Z. Zhao et al., "Deep reinforcement learning for network slicing," arXiv:1805.06591v2, 2018.
[9] M. R. Raza et al., "A slice admission policy based on reinforcement learning for a 5G flexible RAN," ECOC, Sep. 2018.