Using Reinforcement Learning to Allocate and Manage SFC in Cellular Networks

Guto Leoni Santos*, Judith Kelner*, Djamel Sadok*, Patricia Takako Endo†
*Universidade Federal de Pernambuco
Email: {guto.leoni, jk, jamel}@gprt.ufpe.br
†Universidade de Pernambuco
Email: patricia.endo@upe.br

***Abstract*—In this paper, we propose the use of reinforcement learning to deploy a service function chain (SFC) of cellular network service and manage the VNFs operation. We consider that the SFC is deployed by the reinforcement learning agent considering a scenario with distributed data centers, where the virtual network functions (VNFs) are deployed in virtual machines in commodity servers. The VNF management is related to create, delete, and restart the VNFs. The main purpose is to reduce the number of lost packets taking into account the energy consumption of the servers. We use the Proximal Policy Optimization (PPO2) algorithm to implement the agent and preliminary results show that the agent is able to allocate the SFC and manage the VNFs, reducing the number of lost packets.**

***Index Terms*—Service Function Chain, Reinforcement Learning, 5G, Cellular Networks**

I. Introduction

In a cellular network scenario, a service function chain (SFC) may be the core telecom stack [1], and can be deployed in virtual machines located in distributed data centers. This new configuration increases the flexibility of the network and at the same time reduces operating expenditures (OPEX) and capital expenditures (CAPEX) [2]. However, the integration between cellular network and network virtualization function (NFV) to compose new 5G network architectures is not without challenges. It is expected that the performance and availability of applications are maintained with the use of NFV. As highlighted by Gupta et al. [3], maintaining the uptime of cellular networks is one of major issues from the operators' point of view. However, several aspects can impact the services hosted in data centers, such as software failures, misconfiguration, planned and unplanned downtime, and so on [4].

The management of these networks is a complex task. Considering the network status before allocating an SFC can be a promissory way to ensure high level of quality of service [5]. Additionally, a smart strategy need to be employed to guarantee the service high availability (for example, adding redundant VFN of a service) taking into account constraints, such as servers utilization and energy consumption. An approach that is able to learn according with different scenarios could be employ in this scenario, adapting the solution (SFC allocation) according to specific requirements and network conditions.

978-3-903176-31-7

In this paper, we use a reinforcement learning algorithm to allocate and manage SFC automatically in data centers considering a cellular network scenario. We create a scenario based on a data set of Milan region, Italy, that consists of internet activities from users related to 62 days [6]. The scenario considered in this paper is composed of actual cells of the Milan city and distributed data centers to manage the traffic from users (by allocating SFC). The SFC based on services of the Evolved Packet Core (EPC), which is the core network of Long Term Evolution (LTE).

A simulator was developed to represent the users' Internet activities based on the data set; and also to represent eventual failures and repair of the EPC network components. Moreover, we conduct experiments to create clusters of cells based on the Internet activities similarity. Then, the reinforcement learning agent can learn the best strategy to allocate VNFs in servers of distributed data centers, considering that both servers and VNFs may fail during their operation. While new Internet activities arrives in the data center, the agent can create, delete, and restart the VNFs in order to reduce the number of failures. The main purpose is to reduce the number of failures and, consequently, reduce the number of lost packets. An agent based on Proximal Policy Optimization (PPO2) algorithm was trained in the environment and preliminary results show that it is able to allocate the SFC after some steps and manage it controlling the number of lost packets due failures.

II. Material and Methods

A. Scenario

We consider a set of distributed data centers located close to the base stations with VNFs related to the EPC core stack. The EPC is the core of network of LTE, where the control and data traffics are separated and addressed by different entities [7]. We consider that all components of EPC, the serving gateway (SGW), the packet gateway (PGW), the mobility management entity (MME), and the home subscribe server (HSS) will compose the SFC that will be deployed over the distributed data centers. Thus, each EPC component correspond to a VNF which can have redundant instances in order to increase the SFC availability.

We assume that all servers of a data center are connected among them, and all data centers are connected, i.e., any server has connection to all other operational servers in the data

centers. We also consider that each server has a maximum limit of VNFs that can be deployed, representing the limited computation resource of such server. The both servers and VNFs run with a probability failure, which increases over time.

The servers of data centers can eventually fail for different reasons such as problems in storage, memory and processor failures, and so on. Due to these failures, the server and its respective virtual machines become unavailable. In a similar way, when a VNF is created, it can fail due to many reasons (software failures, bugs, unexpected maintenance, and so on). Both server and VNFs can be repaired after a failure, becoming operational again. We also consider that in the case of a server failure, the state of VFNs are maintained, i.e., after the repair, all VNFs are restored and can be process data from cellular users. When all VNFs of the same type (e.g., all SGW VNFs) fails, the SFC becomes unavailable, and Internet activities are lost, impacting the user experience. We consider a real data set of cellular traffic as described in the next subsection.

B. Data set

To represent the Internet traffic data, we used the Telecom Italia data set provided by [6]. This data set provides a multi-source aggregation of telecommunications, news, social networks, and electricity data of two different regions of Italy: the Milan region and the Province of Tentrino.

From the data set, we use Internet activities data of the Milan region, composed of nine municipalities, that is largest metropolitan area in Italy and one of most populous areas of the European Union [8]. The data was collected between 1 November 2013 and 1 January 2014, comprising 62 days.

C. Creating clusters

Different regions of Milan may have different Internet activities patterns. For example, Internet activities in regions further from Milan's center may behave differently. Therefore, we analyze Internet traffic of cells in order to create clusters based on similarity.

We consider the average of Internet activities per day periods to represent the cells, and we used the K-means algorithm to create the clusters [9]. For that, we defined different periods of the day. Each period has four hours starting from 0:00. Afterwards, we calculated the average Internet activity for each period for all cells present in the data set.

We use the Elbow method that varies the number of clusters, k, within a range to find the optimal number of clusters based on the sum of squared error [10]. We vary k from one to 50. Based on the results, we selected $k = 12$, because it is when the sum of squared error stabilises.

The Figure 1 shows the 12 clusters that have similar Internet activities overlaid in the Milan metropolitan area. One can note that the Cluster 1 represents more external areas and the Cluster 6 represents the center of the city of Milan. The other clusters represent different areas within the Milan region.

We can train different agents using reinforcement learning algorithms to learn the best SFC allocation strategy for each cluster of cells, since different clusters have different Internet activities patterns. For example, the agent may allocate new VNFs when the traffic increases, and delete them (or some of them) when the traffic decreases, and as the cluster has different patterns, the the allocation strategy need be different as well.

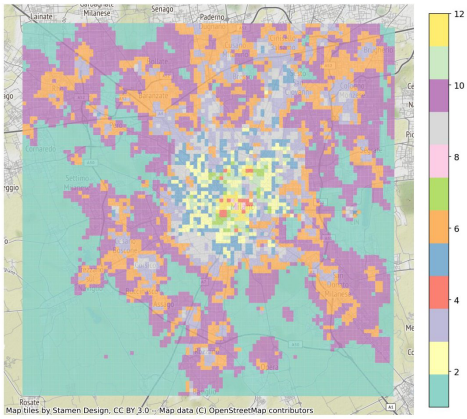

Fig. 1. Overlay of the 12 clusters on a map of the metropolitan area of Milan

III. Reinforcement Learning for SFC Allocation

The goal is to train a reinforcement learning agent to learn how to deploy a SFC taking into consideration the Internet activities using the data set described previously and the overall energy consumption. First of all, we need to define: the environment representation, the action representation, and the reward calculation.

The reinforcement learning agent needs to learn, for each state of the environment, what the action that will give the highest reward. This training consists of agent interactions with the environment over steps, where the agent takes actions that changes the environment state and receives rewards. For each step, we need to provide some relevant information about the environment for the agent. For a given state, the agent tries learn the best action that will return the highest reward. After the training, the agent learns the optimal policy, i.e., a set of actions for each state that will give the highest reward. For more information about the reinforcement learning theory, please see [11].

A. Environment

Our environment definition is composed of two main information: the Internet activities of all cells managed by the data centers, and the number of VNFs already allocated in the servers. To compose a step, we aggregated five minutes of the Internet activities from the Telecom Italia data set, because it is common to have periods when there is no Internet activity in several cells in the data set. We compose a vector with the number of Internet activities of all cells managed by the data centers. This information shows the number of Internet activities that will not be processed if the SFC is not running properly in the data centers, e.g., if there is no one or more VNFs that compose the SFC running.

The other information given to the environment is the number of VNFs running in the servers, which is defined as a vector composed of the number of VNFs of each type present in each server of all data centers. As VNF types, we are considering the four components present in the EPC stack. This vector provides information about the current status of the servers with the purpose for choosing the best server to allocate a new VNF, for example. This vector can also provide information about the energy consumption of overall architecture, which is considered by the agent in the SFC allocation.

B. Actions

Given the environment definition, the agent needs to take actions, and each action has a specific reward for different states. The action representation is a vector of integers as defined in the Equation 1.

$$Action = (a_i, dc_i, s_i, vnf_i) \tag{1}$$

in which, a_i represents the action type that the agent can take, and can be an integer, $a_i \in 1, 2, 3, 4$, where 1 is create a new VNF, 2 delete an existing VNF, 3 restart a VNF, and 4 does not change the environment; dc_i and s_i are, respectively, the data center and server ids where the VNF will be created, and need be defined by the agent; vnf_i is the type of VNF, in our case, the four virtual functions (SGW, PGW, MME, and HSS). Therefore, given this representation, the agent can define actions of different types of VNF specifying the data center and its respective server. In cases where the agent selects the action 4, where the agent does not change the environment, the server and data center ids and the VNF type are not considered.

If the create action is selected, a new VNF will be created in the server for a given data center specified by the agent, since the this server have computational resources available. This resources limit is can be defined by a limited number of VNFs that can be allocated in a server, for example. If the restart action is selected, the probability failure of the VNF type selected is defined as zero and a new failure time is scheduled for this VNF.

It is important highlight that the agent just specifies the type of VNF, but for the restart and delete actions, the agent need to specify what VM will be affected (since there can have redundant VNFs of the same type). When one of these actions (restart or delete) is selected by the agent, we consider that the VNF with the highest fail probability will be affected (deleted or restarted), since our objective is to reduce the failures. For example, if in a given step, the agent takes the action with the specification $\{2, 1, 2, 3\}$, the VNF of type 3 with the highest fail probability, in the server 2 of the DC 1 will be deleted.

C. Reward

After take the action, the agent receives a reward that will define the policy for taking new actions. The main objective of our agent is to reduce the number of VNF failures (and consequently the lost packets) considering the energy consumption of allocating new VNFs. The Equation 2 describes the reward calculation.

$$\begin{aligned} Reward = &- ((1 - sfc_t) * packets_t) \\ &- \sum_{i=0}^{DCs} (energy_i) - repair_t + (sfc_t * f) \end{aligned} \tag{2}$$

in wich $sfc_t \in \{0, 1\}$ represents the SFC status. If all VNFs that composes the SFC (PGW, SGW, MME, and HSS) are operational, then the SFC is complete and $sfc_t = 1$. On the other hand, if there is no VNFs of at least one type running, the SFC is incomplete and $sfc_t = 0$. sfc_t is equals to 0 at the beginning of the system operation, where there is no VNFs allocated, and also if the agent deletes all VNFs of a specific type in all data centers. In this case, the agent receives a negative reward multiplied by the number of Internet activities, $packets_t$, of all cells for a given step t in order to increase the penalty according with the network status. We also apply a penalty for the agent according to the energy consumption of the VFNs allocated. We consider the energy consumption of data center i, $energy_i$, as defined in the Equation 3, where CPU_k and MEM_k are the energy consumption of a CPU and a memory, respectively, in terms of allocate a new VNF at server j. We also apply a little penalty if the agent decide to restart a VNF defined by the variable $restart_t \in \{0, 1\}$, with the purpose not stimulate the agent restart the VNFs frequently.

$$energy_i = \sum_{j=0}^{servers} \sum_{k=0}^{vnfs} CPU_k + MEM_k \tag{3}$$

Finally, the agent receives a positive reward if the SFC is complete ($sfc_t = 1$), that is, if all VNFs that defines a SFC are running in the servers of data centers. The factor f (we consider equals to 100) is applied to adjust the reward for the agent, where the greater the factor, the greater the agent's reward if the SFC is operational.

IV. Preliminary Results

To create the environment described previously, we used the framework Stable Baselines. The Stable Baselines is a fork of OpenAI baselines project with new algorithms, wrappers for preprocessing and multiprocessing, and friendly interfaces to create new agents and environments.

To create the agent, we used the standard implementation of PPO2 algorithm provided by the Stable Baselines framework. The PPO2 combines ideas from Advantage Actor Critic (A2C) [12] and Trust Region Policy Optimization (TRPO) [13] algorithms and outperformed them in complex tasks such as robotic locomotion and Atari game playing. For more details about PPO2, see [14].

We implemented a simulator using Python 3.6 language to simulate the events of incoming Internet activities (based on the data set described in the Subsection II-B) and the failure and repair events of the servers and VNFs. This simulator is used to implement the steps of the environment. At begin of the simulation, the servers are created on the data centers and a failure time is scheduled to happen for each server. When

a VNF is created by the agent, a failure time is scheduled to happen similar to the server. After a failure event, this component becomes unavailable and a repair event of the component is scheduled. After the repair event, the component becomes available again and a failure event is scheduled. The failure and repair events are generated following exponential distributions [15], which are defined by mean time to failure (MTTF) and mean time to repair (MTTR) values, respectively.

We consider a simple scenario to evaluate the PPO2 performance to allocate SFC in the data centers in order to reduce the number of failure and lost packets. Table I summarizes the scenario configuration.

TABLE I
SCENARIO CONFIGURATION

Parameter	Value	Parameter	Value
Number of data centers	10	MTTF VNF	24 hours
Number of servers per data center	5	MTTR VNF	0.033 hours
Maximum number of redundant VNFs	5	CPU energy consumption	40 W
MTTF server	8760 hours	Memory energy consumption	30.72 W
MTTR server	1.667 hours		

We consider the cells of cluster Cluster 5 (see Figure 1) which corresponds to the center of Milan region, and has the highest Internet activity volume. This cluster is composed of 276 cells. We consider 10 data centers to manager these cells, each one with five servers dedicated to the SFC of EPC stack. We distributed them equally through the cells. Due to limited resources, we assume that each server can have a maximum of five VNFs allocated. We also assume that there is a limit of redundant VNFs allocated of the same type in one server, in order to increase the heterogeneity of VNFs in the same server. We consider the energy consumption of CPU and memory in Watts, as show in the Table I.

Due to the stochastic characteristics of our simulation and the agent training process, we carried out 100 experiments, and the results presented in this paper are the average. We defined the number of steps according with the Telecom Italia data set events. As we aggregated the Internet activities for each five minutes and the data set has data about 62 days, we have 8,928 events. We divided these events in 90% for create an environment to train the agent (8,035 steps), and 10% to test the agent (893 steps).

Figure 2 shows the reward achieved by the agent over the steps. One can note that, at the first step, the reward is around -400, and it is justified because at the beginning of the simulation, there is not VNFs allocated yet, then SFC is incomplete and resulting in lost all packets from the cells. But the agent learns quickly that this situation is not suitable and creates VNFs to allocate the whole SFC in the data centers. The agent is able to allocate the entire SFC around step 80, when the reward approaches zero. the reward drops a little, this may be justified by the energy consumption of the allocated NFVs, but the reward starts to grow again soon afterwards. After step 200, the reward stabilizes and remains close to 100 (value of factor f defined in the Equation 2 when the SFC is complete). One can note that the reward stabilizes and has slight osculations (with a little decreasing around the step 250), it happens due to the allocation of new VNFs, that increases the energy consumption and, consequently, receives negative rewards. These oscillations can be caused by little SFC unavailability, due to eventual failures in the servers and VNFs, but the impact is low.

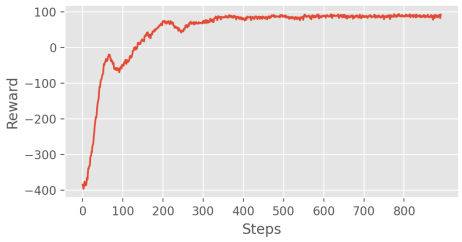

Fig. 2. Reward achieved by agent over time

Figure 3 shows the accumulated lost packets over the steps. Up to step 100, it can be seen that the growth is very fast, given that all packages are being lost, as the SFC is not fully allocated yet. After step 200, the number of packets lost starts to grow slowly. This stabilization can be attributed to the SFC management allocated by the agent, since it can restart the VNFs that are about to fail.

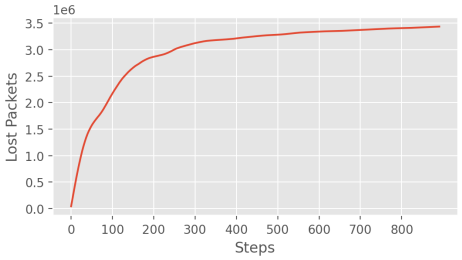

Fig. 3. Accumulated lost packets

V. FINAL CONSIDERATIONS AND NEXT STEPS

In this paper, we carried out simulations using the PPO2 implementation provided by the Stable Baselines algorithms, and preliminary results showed that the PPO2 is able to allocate a complete SFC after some steps and manage the VNFs controlling the lost packets taking into account the energy consumption.

We have several future works to improve the solution presented in this paper. We plan to consider a scenario where many SFCs have to be allocated and managed concurrency. This feature increases the scenario complexity, since SFCs need to be allocated in limited computational resources. We also plan to consider the bandwidth of link between the servers in the VNF placement. New architectures can be considered, where groups of data centers can be considered to manage different regions of a city, for example. Finally, we plan to compare other reinforcement learning algorithms and make a parameter variation to evaluate different configurations.

REFERENCES

[1] D. Bhamare, R. Jain, M. Samaka, and A. Erbad, "A survey on service function chaining," *Journal of Network and Computer Applications*, vol. 75, pp. 138–155, 2016.

[2] J. G. Herrera and J. F. Botero, "Resource allocation in nfv: A comprehensive survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[3] L. Gupta, M. Samaka, R. Jain, A. Erbad, D. Bhamare, and C. Metz, "Colap: A predictive framework for service function chain placement in a multi-cloud environment," in *2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC)*. IEEE, 2017, pp. 1–9.

[4] P. T. Endo, G. L. Santos, D. Rosendo, D. M. Gomes, A. Moreira, J. Kelner, D. Sadok, G. E. Gonçalves, and M. Mahloo, "Minimizing and managing cloud failures," *Computer*, vol. 50, no. 11, pp. 86–90, 2017.

[5] Y. Xiao, Q. Zhang, F. Liu, J. Wang, M. Zhao, Z. Zhang, and J. Zhang, "Nfvdeep: adaptive online service function chain deployment with deep reinforcement learning," in *Proceedings of the International Symposium on Quality of Service*, 2019, pp. 1–10.

[6] G. Barlacchi, M. De Nadai, R. Larcher, A. Casella, C. Chitic, G. Torrisi, F. Antonelli, A. Vespignani, A. Pentland, and B. Lepri, "A multi-source dataset of urban life in the city of milan and the province of trentino," *Scientific data*, vol. 2, p. 150055, 2015.

[7] A. Mohammadkhan, K. Ramakrishnan, A. S. Rajan, and C. Maciocco, "Cleang: A clean-slate epc architecture and controlplane protocol for next generation cellular networks," in *Proceedings of the 2016 ACM Workshop on Cloud-Assisted Networking*, 2016, pp. 31–36.

[8] Eurostat, "Population on 1 january by age groups and sex - cities and greater cities," https://bit.ly/2YqqJeW, 2020, accessed: April, 2020.

[9] P. Arora, S. Varshney *et al.*, "Analysis of k-means and k-medoids algorithm for big data," *Procedia Computer Science*, vol. 78, pp. 507–512, 2016.

[10] M. Syakur, B. Khotimah, E. Rochman, and B. Satoto, "Integration k-means clustering method and elbow method for identification of the best customer profile cluster," in *IOP Conference Series: Materials Science and Engineering*, vol. 336, no. 1. IOP Publishing, 2018, p. 012017.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[12] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[13] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, 2015, pp. 1889–1897.

[14] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[15] E. Andrade, B. Nogueira, R. Matos, G. Callou, and P. Maciel, "Availability modeling and analysis of a disaster-recovery-as-a-service solution," *Computing*, vol. 99, no. 10, pp. 929–954, 2017.