UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

THESE DE DOCTORAT DE

L'UNIVERSITE DE RENNES 1
COMUE UNIVERSITE BRETAGNE LOIRE

ECOLE DOCTORALE N° 601
Mathématiques et Sciences et Technologies de l'Information et de la Communication
Spécialité : *Informatique*

Par

« Imad ALAWE »

« Architectures evaluation and dynamic scaling for 5G mobile core networks »

Thèse présentée et soutenue à « Rennes », le « 21 Novembre 2018 »
Unité de recherche : IRISA
Thèse N° : (8)

Rapporteurs avant soutenance :

Chaouchi, Hakima, Professeur, Telecom Sud Paris
Turletti, Thierry, Directeur de Recherche, INRIA Sophia Antipolis – Méditerranée

Composition du Jury :

Président :	Turletti, Thierry, Directeur de Recherche, INRIA Sophia Antipolis – Méditerranée
Rapporteurs:	Chaouchi, Hakima, Professeur, Telecom Sud Paris Turletti, Thierry, Directeur de Recherche, INRIA Sophia Antipolis – Méditerranée
Examinateurs :	Hadjadj-Aoul, Yassine, Maître de conférences, Université de Rennes 1 Ksentini, Adlen, Maître de conférences (HDR), Eurecom Mokdad, Lynda, Professeur, Université Paris-Est, Val de Marne Nguyen, Thi-Mai-Trang, Maître de conférences, Sorbonne Université - LIP6
Dir. de thèse :	Viho, César, Professeur, Université de Rennes 1
Co-dir. de thèse :	Bertin, Philippe, Ingénieur de Recherche, Orange-Labs, Cesson-Sévigné

Invité(s)
Mme Amélie Kerbellec, Ingénieur chez TDF.

ACKNOWLEDGEMENT

First of all I would like to thank my thesis director, Cesar Viho. I am deeply grateful for the advice and support that I have received from him over the past three years.

My most sincere thanks goes to Yassine Hadjadj-Aoul, Adlen Ksentini, and Philippe Bertin who supervised me throughout this thesis and contributed greatly to my training on mobile core networks.

I would like to thank also Christian Nieps, Davy Darche and Amelie Kerbellec from TDF for their support during the last three years.

In addition, I would like to thank from IRT b<>com our lab chief Olivier Choisy, our domain chief Michel Corriou and our director Bertrand Guilbaud. Also a sincere thanks to our project team for supporting me for the last three years where I spent 80% of my thesis time. A special thank to Carole Bonan, Frederic Leroy, Franck Messaoudi, Cedric Morin, Cao-Thanh Phan, and Frederic Sene.

I would like also to thank all my colleagues from Dionysos team at IRISA for also hosting me at their lab and their collaboration during my thesis.

I would like to express my sincere thanks to the committee members, Hakima Chaouchi, Lynda Mokdad, Thi-Mai-Trang Nguyen, and Thierry Turletti for accepting to evaluate this work and to be part of the jury.

I would also to thanks my friends, Saidou Bangoura, Matthieu Kanj, Ahmed Mheich, Ajit Rai, and Ali Zeineddine for their support.

Finally, the deepest gratitude and heartiest thanks for my parents who supported me not only during the last three years but also since I was born.

TABLE OF CONTENTS

Résumé En Français **15**

1 Introduction **17**
1.1 Motivations 17
1.2 Contributions of the Thesis 18
1.3 Organization of the Manuscript 19

2 Mobile core network evolution towards 5G **21**
2.1 Introduction 21
2.2 5G: performance expectations and requirements 22
2.3 5G Key Enablers 23
2.3.1 Software Defined Networking at a glance 23
2.3.2 Network Functions Virtualisation 25
2.3.3 Network Slicing 26
2.4 Mobile Core Network towards 5G 27
2.4.1 Evolved Packet Core in a glance 27
2.4.2 Full Virtualisation 29
2.4.3 Control Plane Virtualisation 30
2.4.4 Hybrid / Mix Virtualisation 32
2.4.5 Community research work summary 33
2.5 New Generation Core for 5G 34
2.6 Conclusion 37

3 On Evaluating Different Trends for Virtualised and SDN-ready Mobile Network **39**
3.1 Introduction 39
3.2 Related work 40
3.3 Benchmark Architectures 42
3.3.1 Virtual EPC (vEPC) 42

3.3.2 SDN EPC 42
3.3.3 Enhanced EPC 45
3.4 Test scenarios and methodology 45
3.5 Performance evaluation 47
3.5.1 Results analysis 47
3.5.2 Discussion 50
3.6 Conclusion 51

4 Congestion Control In New Generation Core Network of 5G 53
4.1 Introduction 53
4.2 Related Work 54
4.3 Architecture 55
4.3.1 Traffic steering 56
4.3.2 Scale In/Out 57
4.4 AMF Balancing and Scaling Proposal 58
4.4.1 Model description 58
4.4.2 Dynamic AMF Load Balancing 61
4.4.3 Dynamic AMF Scaling 62
4.5 Model Evaluation and Results Analysis 62
4.5.1 Test Scenarios 62
4.5.2 Results Analysis 63
4.6 Conclusion 66

5 Congestion Avoidance In New Generation Core Network of 5G 67
5.1 Introduction 67
5.2 Related Work 68
5.3 Architecture 70
5.4 Basics on Neural Networks 70
5.4.1 DataSet 70
5.4.2 Neural Network Types 70
5.5 ML-based scalable mechanism for 5G core network 73
5.5.1 Traffic forecasting: Deep learning-based approach 73
5.5.2 AMF Balancing and Scaling Proposal 76
5.5.3 Models Evaluation and Results Analysis 79

5.6 Neural networks traffic load estimation for resources' scaling in 5G 82
5.6.1 Generalizing neural nets and accelerating the training 82
5.6.2 Models Evaluation and Results Analysis 84
5.6.3 Results and Analysis 85
5.7 Conclusion 88

6 Conclusion **91**
6.1 Introduction 91
6.2 Contributions of the thesis 92
6.3 Perspectives 93

Publications from the thesis **95**

Bibliography **97**

Acronyms **107**

LIST OF FIGURES

2.1 5G Expectations and Applications 22
2.2 SDN architecture 23
2.3 NFV MANO Architecture 25
2.4 Network Slicing Example 27
2.5 Evolved Packet Core Architecture 28
2.6 Fully virtualized EPC architecture 29
2.7 Control Plane Virtualisation EPC architecture 30
2.8 completely virtualised EPC architecture 33
2.9 Mobile Core Network architectures: Towards 5G Classification 34
2.10 Service oriented 5G Core network 36

3.1 3GPP TR 23.714 Split Architecture. 41
3.2 EPC with SDN integration. 43
3.3 UE Attach Call Flow with SDN integration. 44
3.4 Attach duration over (a) control plane, (b) control and data plane. 47
3.5 UE accepted ratio over the C plane. 48

4.1 Architecture proposal for 5G 56
4.2 System Model 59
4.3 Evaluation of the Adaptation Of the Control Model 64
4.4 Evaluation of the control model Behavior vs different arrival Load 65

5.1 DNN architecture 71
5.2 Long Short Term Memory Cell Architecture 71
5.3 CNN-LSTM Architecture 73
5.4 Reality vs NN prediction 76
5.5 Proposal Logic 77

5.6 Evolution of AMF instances number with control traffic Requests (a) threshold-based solution (b) DNN-based solution (c) RNN-based solution compared to the evolution of AMF instances load with AMF instances number (d) threshold-based solution (e) DNN-based solution (f) RNN-based solution 80
5.7 AMF instances number vs Blocked traffic Requests (a) threshold-based solution (b) DNN-based solution (c) RNN-based solution compared to the evolution of the UEs attach duration (d) threshold-based solution (e) DNN-based solution (f) RNN-based solution 81
5.8 Testing data used to test the Neural Networks 84
5.9 Evolution of prediction distance (a) and training phase duration (b) vs number of epochs in training phase 86
5.10 Evolution of prediction distance (a) and training phase duration (b) vs number of clusters in training phase 87
5.11 Evolution of prediction distance (a) and training phase duration (b) vs number of boost in training phase 88

LIST OF TABLES

3.1 Configuration template of the EPC modules. 45
3.2 Testbed KPIs 46
3.3 MME CPU usage over pattern tests. 49

RESUMÉ EN FRANÇAIS

Les réseaux de communications mobiles sont largement déployés et utilisés dans le monde entier. Développés à partir des années 1980, ils sont en constante évolution depuis. Ils sont utilisés dans de nombreux domaines, comme le domaine médical, commercial, éducatif, militaire, etc. Par conséquent, le nombre des objets connectés augmente de manière exponentielle. Jusqu'à 28 milliards d'appareils seront connectés au réseau 5G d'ici 2021, contre 17 milliards en 2016 [1]. Ce grand nombre d'appareils connectés est dû à l'apparition de nouveaux services, tels que l'Internet des Objets (Internet Of Things (IOT)), la santé connectée (e-Health), etc. Afin de répondre à ce volume élevé de trafic, la communauté de standardisation des télécoms a commencé à concevoir et développer la prochaine génération de système de communication mobile (5G) basée sur les solutions de réseaux logiciels (Software Defined Networking (SDN)), la virtualisation des fonctions réseaux (Network Functions Virtualisation (NFV)) et le cloud computing. Ces futurs systèmes 5G visent non seulement à augmenter la bande passante, mais également à élaborer une nouvelle architecture de réseau mobile complète, très flexible, dynamique, évolutive et conçue à la demande pour répondre aux exigences des nouvelles applications et services 5G. Pour spécifier les attentes de la 5G, l'organisme de standardisation 3rd Generation Partnership Project (3GPP) ainsi que le groupe Next Generation Mobile Networks (NGMN) ont produit des documents résumant plusieurs services ainsi que des scénarios d'utilisation [2] [3]. Parmi ces services et scénarios, trois catégories émergent: (i) le haut débit mobile évolué; (ii) la communication massive de type machine (Machine Type Communication (MTC)); (iii) la latence ultra faible. Du fait que ces trois catégories correspondent à des attentes différentes en termes de connectivité réseau et d'indicateurs de performance (Key Performance Indicator (KPI)), l'architecture 5G envisagée se doit d'être programmable et capable de supporter les trois catégories de services en partageant la même infrastructure physique, tout en garantissant les différents KPI (c.-à-d. disponiblité de bande passante, fiabilité, faible latence). Pour répondre aux besoins de ces services hétérogènes, tout en utilisant la même infrastructure, les solutions de virtualisation du réseau, basées sur SDN, (NFV) et le cloud computing, s'imposent. Dans ce con-

texte, différentes initiatives issues de l'industrie et de la communauté de recherche proposent déjà des moyens d'intégrer la technologie NFV et le SDN dans le cœur mobile 4G. La plupart des propositions convergent vers une orientation commune, qui consiste à séparer les fonctions du plan de contrôle et du plan de données des entités fonctionnelles du cœur de réseau (Evolved Packet Core (EPC)) (à savoir Mobility and Management Entity (MME) et Serving GateWay/PDN GateWay User Plane functions (S/PGW)). Mais, il demeure des divergences, en particulier dans le choix de l'utilisation ou non de contrôleur SDN dans la 5G.

La première contribution de cette thèse consiste à évaluer les différentes visions, pour le futur cœur 5G, à travers leur mise en oeuvre dans une plate-forme de tests. Ainsi, nous avons identifié les trois approches principales: (i) EPC virtuel (Virtual EPC (vEPC)); (ii) EPC virtuel amélioré (Enhanced Virtual EPC (evEPC)); et (iii) EPC intégrant un contrôleur SDN (SDNEPC). L'approche vEPC consiste à virtualiser les composants de l'EPC 4G (c'est-à-dire MME, Serving GateWay (SGW), PDN GateWay (PGW)) et à les exécuter sur des machines virtuelles (Virtual Machine (VM)) dans le cloud. Dans cette approche, aucune séparation pour le plan de contrôle et le plan de données n'est mentionnée. La seconde approche (evEPC) adopte, en plus de la virtualisation des composants EPC, la séparation des fonctions de contrôle et du plan de données. Cette vision est choisie par le groupe 3GPP Control and User Plane Separation (CUPS) [4], qui fournit des spécifications détaillant les interfaces d'architecture ainsi que les fonctions appartenant au plan de données et celles appartenant au plan de contrôle. Ainsi, la SGW et la PGW sont divisées en: (i) Serving GateWay Control Plane functions (SGW-C) et PDN GateWay Control Plane functions (PGW-C) pour gérer les fonctions du plan de contrôle; et (ii) Serving GateWay User Plane functions (SGW-U) et PDN GateWay User Plane functions (PGW-U) pour gérer le trafic du plan utilisateur. Le 3GPP CUPS ne spécifie pas la présence d'un contrôleur SDN entre le Serving GateWay/PDN GateWay Control Plane functions (S/PGW-C) et le Serving GateWay/PDN GateWay User Plane functions (S/PGW-U). De nouvelles interfaces sont spécifiées qui permettent aux entités SP / GW-C de contrôler directement les tables de traitement et de commutation de flux de la S/PGW-U. Enfin, la troisième approche (SDNEPC) peut être sub-divisée suivant deux visions: (i) une vision ne recherchant pas la compatiblité avec les protocoles standards du 3GPP; et (ii) une vision compatible 3GPP. La première propose de repenser l'architecture actuelle par une architecture de réseau IP géré par un Contrôleur SDN et évitant de se reposer sur les

tunnels GPRS Tunneling Protocol (GTP) [5] pour l'acheminement des flux utilisateurs. Toutes les fonctions du plan de contrôle (authentification, gestion de la mobilité, signalisation, contrôle du plan de données) sont exécutées en tant qu'applications réseau au-dessus du contrôleur SDN, via ses Application Programming Interface (API) nord. Pour la gestion du trafic du plan de données, des règles sont poussées par le contrôleur SDN via une interface sud en utilisant, par exemple, le protocole Openflow OpenFlow Protocol (OF) [6]. Le plan de données repose ainsi sur des commutateurs compatibles avec le protocole OF. La seconde vision propose de maintenir la compatibilité avec l'architecture 3GPP en séparant les fonctions de données et de contrôle, tout en conservant l'utilisation du protocole GTP pour gérer et acheminer le trafic du plan de données.

Les objectifs de cette contribution sont doubles. Premièrement, construire une plate-forme de test pour évaluer les performances de l'approche vEPC; ce qui donne un aperçu des performances d'un EPC entièrement virtualisé et de la manière dont les instances virtualisées du MME, S/PGW, pourraient être dimensionnées en termes de ressources CPU et mémoire. Deuxièmement, étendre la plate-forme de test avec les approches evEPC et SDNEPC et comparer les performances des trois solutions. Pour atteindre ces objectifs, une plate-forme de test a été créée en utilisant uniquement des solutions open source. Pour les entités mobiles (à savoir eNodeB, User Equipment (UE) et EPC (S/PGW, MME, Home Subscriber Server (HSS))), nous avons utilisé OpenAirInterface5g (OAI) [7] [8]; alors que OpenDayLight (ODL) [9] et Open Virtual Switch (OVS) [10] ont été utilisés respectivement comme contrôleur SDN et comme commutateur OF (S-P / GW-U). Nous avons modifié OAI pour émuler l'attachement d'environ 900 UE, afin de stresser (monter en charge) les entités EPC. Concernant les indicateurs de performance mesurés, nous nous sommes concentrés sur la durée d'attachement, le nombre d'UE connectés avec succès au plan de contrôle (principalement la couche Non-Access Stratum (NAS)) pendant une période donnée et le temps nécessaire pour établir le tunnel GTP du plan de données (tunnel par défaut) pour les UE. Les résultats obtenus nous permettent de tirer des conclusions intéressantes sur le dimensionnement de la version virtualisée d'EPC, en termes de capacité du processeur, ainsi que sur la flexibilité et l'évolution apportée par SDN et NFV dans les réseaux mobiles. En complément, les résultats de tests ont mis en évidence un problème de performance concernant le MME. Comme le MME est un point d'accès unique au plan de contrôle du cœur du réseau 4G, il peut être facilement surchargé lorsque le

nombre de terminaux connectés est en augmentation constante avec la 5G. De toute évidence, il est nécessaire de prévoir un mécanisme pour résoudre ce problème, ce qui nous amène naturellement à la seconde contribution de cette thèse.

Dans cette deuxième contribution, nous nous intéressons principalement à la fonction de gestion des accès et de la mobilité (Access and Mobility Function (AMF)), à la fonction de gestion de session (Session Management Function (SMF)) et à la gestion unifiée des données (Unified Data Management (UDM)) du nouveau cœur de réseau 5G. Ces trois fonctions étant issues du composant MME de la 4G. Dans cette contribution, nous proposons un algorithme de mise à l'échelle basé sur la théorie de contrôle pour les instances AMF; ce qui permet: (i) de contrôler la charge de chaque AMF en envoyant des requêtes à chaque instance AMF en fonction de sa charge et en fonction de la charge totale du système; (ii) de tirer parti de NFV pour dimensionner le système de manière dynamique, en déployant de nouvelles instances AMF ("scale out") ou en supprimant des instances AMF ("scale in"), en fonction de la charge. La solution proposée est testée et vérifiée à l'aide de simulations, et ses résultats sont comparés à un modèle utilisant la répartition probabiliste des procédures et la moyenne mobile pondérée exponentiellement (Exponentially Weighted Moving Average (EWMA)) pour la mise à l'échelle / dans les instances de l'AMF. Nous avons montré que la conception de ce contrôle permet d'économiser des ressources en augmentant et en réduisant la capacité de l'AMF selon les besoins. Par ailleurs, des adaptations sont proposées pour pouvoir déployer l'algorithme dans l'architecture New Generation Core (NGC) selon les normes 3GPP.

Bien que le contrôle de la congestion dans le réseau cœur 5G soit bon, éviter la congestion serait encore mieux. Pour cela dans la troisième contribution de la thèse nous proposons un nouveau mécanisme pour adapter les ressources du réseau cœur 5G en anticipant les évolutions de la charge de trafic grâce à la prédiction via les techniques d'apprentissage automatique. La prédiction de la charge de trafic est obtenue en utilisant et en entraînant un réseau neuronal sur un ensemble de données réelles d'arrivée du trafic de contrôle dans un réseau mobile. La proposition est testée via un simulateur à événements discrets utilisant un ensemble de données de charge de trafic fourni par un opérateur de réseau. Les résultats montrent clairement une meilleure robustesse avec notre proposition par rapport à une technique de mise à l'échelle classique basée sur un seuil. De plus, nous proposons une nouvelle solution pour généraliser les réseaux de neurones tout en accélérant le processus d'apprentissage

en utilisant la classification K-means et une méthode de Monte-Carlo. Nous avons comparé plusieurs types de réseaux de neurones en utilisant les données réelles d'un opérateur afin de comparer leur efficacité dans la prévision de la charge du réseau à venir pour un provisionnement dynamique et proactif des ressources. La solution proposée permet d'obtenir de très bonnes prévisions de l'évolution du trafic tout en réduisant de 50% le temps nécessaire à la phase d'apprentissage.

Pour conclure, de nouveaux services et de nouveaux types d'appareils connectés apparaîtront avec l'émergence de la 5G. Ces services et appareils connectés ont des exigences multiples et créent ainsi de nouveaux défis pour la conception de la 5G. La 5G est censée garantir des services qui ont besoin d'une latence ultra faible, d'une bande passante élevée et d'un besoin massif de connectivité. Afin de répondre à ces exigences en partageant les mêmes infrastructures physiques, de multiples facteurs clés devraient être envisagés dans le développement et la conception de la 5G. Le SDN, le NFV et le découpage réseaux (Network Slicing) sont les éléments clés qui permettent de répondre à ces nouvelles exigences. Dans cette thèse, le travail se concentre sur les performances du réseau central mobile et les solutions d'évolutivité dynamique des composants du cœur du réseau mobile, dit nouvelle génération. Comme le nombre d'appareils connectés tend à augmenter, l'évolutivité dynamique devrait être envisagée pour le NGC. Des benchmarks de performance ont été réalisés et de multiples algorithmes ont été proposés afin de construire un mécanisme de mise à l'échelle dynamique.

CHAPTER 1

INTRODUCTION

1.1 Motivations

Upcoming 5G systems aim to not only increase the user throughput, but also bring into light a complete new mobile network architecture, which should be very flexible, dynamic, scalable and built on demand to fulfill the requirements of the new envisioned 5G services and applications. To clearly specify the expectations behind 5G systems, the 3GPP as well as the NGMN have produced some documents summarizing several envisioned services and use-cases [2] [3]. From these services and use-cases three categories emerge: (i) Enhanced Mobile Broadband; (ii) Machine Type Communication (MTC); (iii) Ultra-low latency. Stemming from the fact that these three categories have different expectations in terms of network connectivity and KPI, the envisioned 5G architecture should be programmable and able to run these three categories by using (sharing) the same physical infrastructure. Indeed, to fulfill heterogeneous services' requirements, while using the same infrastructure, network virtualisation solutions, based on SDN, NFV and Cloud Computing, are mandatory.

In addition, according to recent forecast, up to 28 billion devices will be connected to 5G by 2021, in front of 17 billions in 2016 [1]. Hence, to support the expecting high number of devices, 3GPP has rethought the Core network architecture aiming at being more flexible and scalable. The new architecture, namely New Generation Core (NGC) [11], addresses scalability and flexibility by introducing more modular Network Functions (NF) to compose the control plane service, which could also rely on network virtualisation via NFV.

Thus, the main challenge of running core network elements in a virtualised environment is the Virtual Network Functions (VNF) dimensioning, in terms of resources, which highly impacts performances. Indeed, the virtualised core network elements should perform similarly to their hardware version; that is, supporting the same Quality of Service (QoS), in terms of supported number of users, provided data rate, etc. To

reach the same performance as the hardware version, over provisioning the allocated resources for a VNF could be envisioned but it might be costly. Alternatively, adaptive provisioning of virtual resources could represent an interesting solution where the VNF is over sized (by increasing the number of CPUs or by multiplying the number of VMs instances of the VNF) when traffic load is growing, and down sized when traffic is decreasing. Such a solution is known as a scale-up/down (if the number of CPU is increased/decreased) or scale-out/in process (if the number of VMs serving the same VNF is augmented/reduced), which is highly used in cloud computing. To scale VNF resources, there is a need of a solution taking scaling decisions. While in cloud computing the decision is mainly based on system level information, like consumed CPU or memory, which is obtained via the virtualisation platform, mobile networks require service level information, like the number of users connected to the system or the traffic load, which indicate if the VNF is overloaded or not. Indeed, knowing what is acceptable for a service is not directly related to the load of a system, in terms of memory or CPU, but to the requirements of the service itself.

1.2 Contributions of the Thesis

The objective of this thesis is to address the challenges related to the NGC Network architectures, by evaluating the performances of the virtualised architectures and solving scalability issues by introducing novel strategies. The contributions of the thesis can be regrouped around two main axes as follows:

1. Evaluating different mobile core architectures visions. This contribution was published in [12]. In this comparative analysis, we used a testbed platform (based on open source tools) to evaluate the performances of: (i) an vEPC; (ii) a evEPC, where data and control planes are decoupled; (iii) an SDNEPC where an SDN controller is deployed. We were interested on metrics regarding the control plane and data plane performances. The obtained results allowed us to draw very interesting conclusions about the dimensioning of the virtualised version of EPC, in terms of CPU capacity, and the flexibility as well as the scalability brought by SDN and NFV in mobile networks.

2. Scaling and load balancing strategies for New Generation Mobile Core network. This axis may be split into two main ideas:

(a) Load balancing and reactive scaling based on static thresholds. In [13], we proposed a scaling algorithm based on control theory for the Access and Mobility Function (AMF) instances, which allows: (i) controlling the load of each AMF by dispatching the requests based on each AMF load and the whole system load; (ii) taking benefit from NFV to dimension the system dynamically, by deploying new AMF instances (scale out) or remove AMF instances (scale in), according to the traffic load. The proposed solution was tested and verified using computer simulations, and its results were compared with a model using probabilistic procedure dispatching and EWMA for scaling out/in the AMF instances. Besides, procedures were proposed to be able to deploy the algorithm in the NGC architecture following the 3GPP standards.

(b) Proactive scaling based on Machine learning and more precisely on Deep Neural Networks. This axis led to multiple publications: [14], [15], [16], and [17]. Mostly we analyzed the most effective techniques for predicting the traffic evolution of a mobile network. We were particularly interested in the possibility of generalizing prediction techniques to allow them to effectively predict traffic evolution even when the scale of traffic changes. To that purpose, we introduced a novel technique allowing to generalize the neural network while accelerating significantly the training phase using K-means clustering and a Monte-Carlo method. We applied our technique to three types of neural networks: Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) cell; Convolutional Neural Network (CNN); and CNN combined with LSTM Neural Network. Finally we did a benchmark to list the effects of parameters' variation (i.e. number of clusters, epochs, number of boosts, etc.) on the prediction efficiency for each of the neural network types listed above.

1.3 Organization of the Manuscript

The thesis is organised as following:

Chapter 2 introduces a general context around 5G. It introduces the requirements and the expectations of 5G, then it lists the key enablers technologies like NFV and SDN allowing the realization of 5G. Then, the EPC of 4G is presented in a glance.

Afterwards, a small survey on the works done by the research community on the evolution from 4G towards 5G of the core network architectures is proposed. A diagram summarizing the different works is drawn. Finally the work of the standardization group (3GPP) around 5G core Network (NGC) is presented.

Chapter 3 presents the first contribution of this thesis. It starts by introducing the context of the proposed architectures towards 5G as envisioned by 3GPP and the research community, and the goal of the contribution; which is benchmarking different proposed architectures. In addition, this chapter presents and details the contribution, the test-bed, the test scenarios and the obtained results. Finally, the chapter is concluded by interesting remarks which allow to introduce the next contribution.

Chapter 4 presents the second contribution, which aims to propose a control congestion solution in a mobile core network based on the new architecture of 5G. This chapter starts by introducing the context of 5G and congestion risks. Then the architecture context of the NGC used to test the contribution is presented. Following that, the algorithm and the mechanism developed for this contribution are detailed before introducing the testing scenarios. Finally, the tests results are discussed, then, followed by a conclusion to the chapter.

Chapter 5 introduces the third and the final contribution of this thesis which consists in a congestion avoidance solution in a 5G mobile core network. Firstly, the context of the contribution as well as the scientific materials are presented. Then, the first part of the contribution, that consists on predicting the upcoming load in a mobile core network is presented. Following that, a use case of this contribution is presented as well as scenarios of test using discrete event simulations. The second part of the contribution, generalizing the neural network and accelerating the training phase of a neural network for upcoming load prediction in 5G NGC, is then presented. The algorithm and the principles of this contribution are listed before introducing the testing simulations. The results are discussed and a conclusion is drawn at the end of this chapter.

Finally chapter 6, concludes this thesis and presents the research perspectives and axes.

CHAPTER 2

MOBILE CORE NETWORK EVOLUTION TOWARDS 5G

2.1 Introduction

Mobile communication systems are widely deployed and used over the globe. They have been developed since 1980s and are in continuous evolution till today. They are used in many fields i.e. medical, business, education, military etc.

Hence, the number of connected devices is increasing exponentially. Up to 28 billion devices will be connected to 5G core network by 2021, in front of 17 billions in 2016 [1]. This massive number of connected devices is due to the emergence of new services i.e. IOT [18], Connected-Health (e-Health) etc. This will generate voluminous data traffic yielding to more investment from network operators, in term of capacity, in order to accommodate this high volume of traffic. Telco community has started to think and work on the next generation of mobile communication system (5G) while using a shared, in term of services, physical infrastructure. Aiming at fulfilling these needs, network Virtualisation solutions, based on SDN, NFV and Cloud Computing, are interesting candidates to explore.

In this chapter, a global view on the expectation of 5G in terms of performance and requirements will be introduced. A review about the EPC is presented. In addition, key enablers for 5G in term of technologies will be introduced. Finally, the community effort regarding the evolution of the mobile core architecture from 4G toward 5G will be discussed.

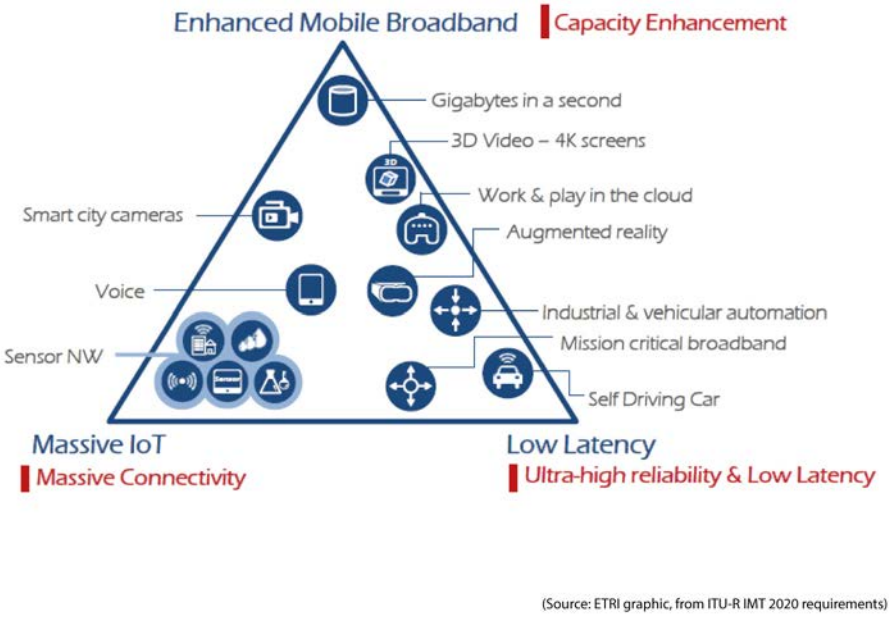


Figure 2.1: 5G Expectations and Applications

2.2 5G: performance expectations and requirements

Following both technological and lifestyle evolution, 5G is expected to offer multiple services that demands different requirements in terms of QoS. Thus a single shared physical infrastructure should answer the needs of those different types of services. From [19], Figure 2.1 illustrates the expected services to be offered by 5G and their requirements. The main requirements can be split into three main categories:

- Capacity Enhancement
- Ultra-High reliability and Low Latency
- Massive Connectivity

The capacity enhancement is expected for Enhanced Mobile Broadband applications that demand a high bandwidth. Hence in 5G we expect offering Gigabytes per second for 3D Video, 4K screens and Augmented Reality. Also Massive connectivity is imagined for the communication needs of multiple (billions of) IoT devices (massive number of IoT devices). As sensors send small data (in size) while connecting by batches to the network, they require massive connectivity support over the control plane. Finally,

self-Driving cars and Mission critical broadband will require Low latency networks as they are time-constrained applications.

Multiple key enablers emerged recently at the core network allowing to answer the demand [20]. We can cite, SDN, NFV and also network slicing. In the following section a brief introduction of those key enablers will be introduced.

2.3 5G Key Enablers

Following the introduction of 5G requirements and expectations, we will briefly introduce some of the most important technological key enablers of 5G systems.

2.3.1 Software Defined Networking at a glance

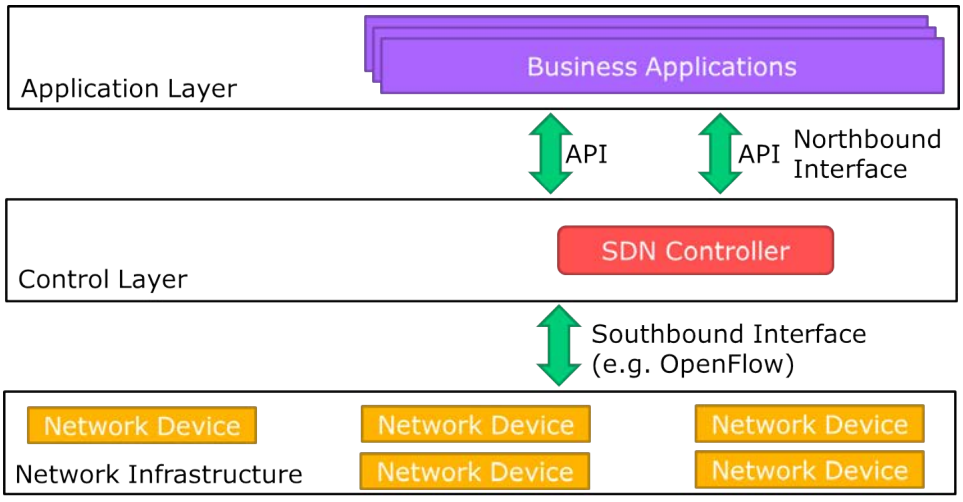


Figure 2.2: SDN architecture

SDN is a network paradigm dedicated to simplify networking and enables the emergence of new applications. In the SDN paradigm, the network is separated into three main layers illustrated in Figure 2.2. SDN main goals are:

- Control plane and data plane separation. Network devices become simple forwarding devices.
- Flow based forwarding decisions, instead of destination based.

- Centralized control logic in an external entity (the SDN controller).

- Programmable network through software applications running on top of the SDN Controller.

In order to achieve those goals, the SDN paradigm can be depicted into three main layers as follows from bottom to top:

1. Layer 1: network devices interconnected between each others providing the network infrastructure;

2. Layer 2: SDN controller, controlling the network infrastructure by pushing configurations and forwarding rules to it;

3. Layer3: Applications hosting all the control logic to be applied on the network (Network Operating System (NOS) brain).

This separation in SDN gives the opportunity to answer the 5G challenges by decoupling the control plane from the data plane and thus moving the control logic into the NOS. The SDN controller will use the provided network APIs to the application layer to decide which logic to apply and so pushing the adequate configurations into the network devices. It is worth noting that making the network programmable increases network reliability and security, automates management, uniforms policy enforcement and reduces configuration errors. Beside the architecture, in SDN the communication between the layers is key. Accordingly, two important interfaces are identified for the SDN controller: the northbound interface and the southbound interface. The northbound interface of the SDN controller allows communications between the controller and applications. It is not standardized yet but the most used interfaces supported by existing controllers are based on Representational State Transfer (REST) APIs. For the southbound interface, a standard has been defined by the Open Networking Group (ONF) allowing the communication between the SDN controller and the network devices: the OF protocol. Beside the OF protocol, other protocols can also be used like NETCONF, Simple Network Management Protocol (SNMP), etc. Readers interested in more details about SDN may refer to [21] and [22].

2.3.2 Network Functions Virtualisation

This key enabler is expected to offer lot of gains to Operators. To start NFV is, as it names tells, virtualizing network functions and services that usually run on proprietary, and dedicated hardware. Thus a given Network Function can be run in a virtual machine in the cloud or locally on a general purpose machine. The expected benefits of this key enabler can be listed as defined in [23]:

- Reduce Capital Expenditure (CapEx): reducing the need to purchase purpose-built hardware and supporting pay-as-you-grow models to eliminate wasteful over-provisioning.
- Reduce Operational Expenditure (OpEx): reducing space, power and cooling requirements of equipment and simplifying the roll out and management of network services.
- Accelerate Time-to-Market: reducing the time to deploy new networking services to support changing business requirements, seize new market opportunities and improve return on investment of new services. Also it lowers the risks associated with rolling out new services, allowing providers to easily trial and evolve services to determine what best meets the needs of customers.
- Deliver Agility and Flexibility: quickly scale up or down services to address changing demands; support innovation by enabling services to be delivered via software on any industry-standard server hardware.

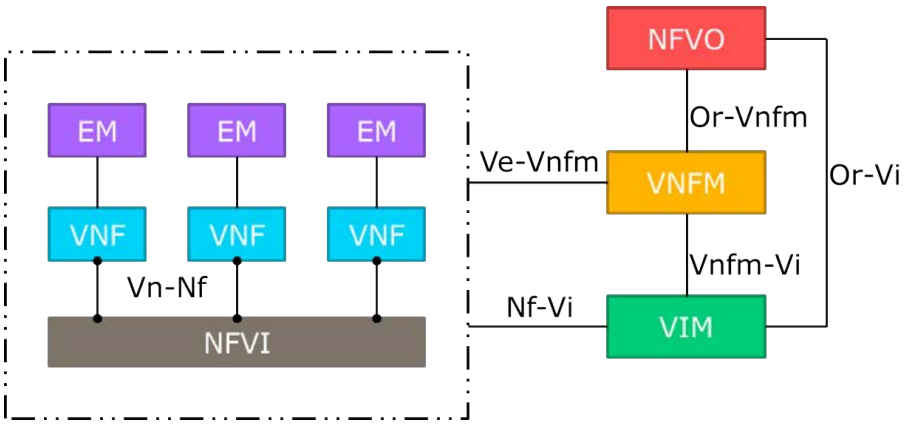


Figure 2.3: NFV MANO Architecture

It is worth mentioning that the European Telecommunications Standards Institute (ETSI) group is working on an architecture (framework) allowing to manage NFVs: the Management And Orchestration (MANO) architecture [24], illustrated briefly in Figure 2.3, which is mainly composed of three blocks:

- NFV Orchestrator (NFVO): responsible for on-boarding of new Network Services (NS) and VNF packages; NS lifecycle management; global resource management; validation and authorization of Network Functions Virtualisation Infrastructure (NFVI) resource requests
- VNF Manager (VNFM): Oversees lifecycle management of VNF instances; coordination and adaptation role for configuration and event reporting between NFVI and EM
- Virtualised Infrastructure Manager (VIM): Controls and manages the NFVI compute, storage, and network resources
- Element Manager (EM): Provides network management of the VNFs and the physical network elements
- NFVI: It is the hardware and software components that build the environment where VNFs are deployed

2.3.3 Network Slicing

As 5G is expected to offer multiple services to several vertical business entities, while using the same physical infrastructure, with different requirements in term of QoS, network slicing is anticipated as being a key enabler. Network slicing, as depicted in Figure 2.4 and following [25], consists on building logical networks on a virtualised single network infrastructure. The programmability of a network service is then realized in two steps: (i) selecting an appropriate network slice for the service and (ii) dynamically programming the network service within the selected network slice. This key enabler is beyond the scope of the thesis. Readers interested in network slicing for next generation network may refer to [26], [27] and [28].

Note, however, that this concept is not conflicting with the proposed approach. Indeed, the EPC core and its scalability could be itself represented by a slice in future networks.

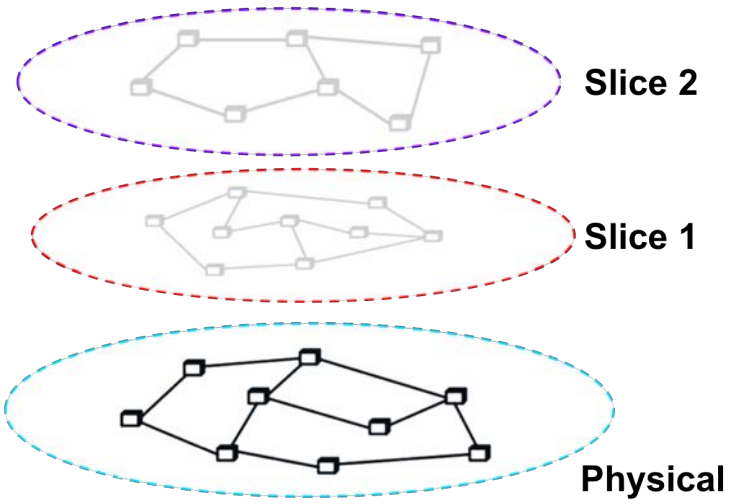


Figure 2.4: Network Slicing Example

2.4 Mobile Core Network towards 5G

As the 5G expectations and requirements are now listed, we direct our focus, in the following, on the EPC architecture, which is briefly introduced, and the research community efforts for evolving towards 5G from an architectural point of view.

2.4.1 Evolved Packet Core in a glance

The 4th generation of mobile networks [29] emerged in 2008 offering new services as Video call conference, mobile TV, mobile gaming, etc. The components of the EPC [30] are depicted on figure 2.5 and briefly introduced in the following:

- HSS is a database that contains user-related and subscriber-related information. It also provides support functions in mobility management, call and session setup, user authentication and access authorization.
- MME deals with the control plane. It handles the signalling related to mobility and security for Evolved Universal Terrestrial Radio Access Network (EUTRAN) access. The MME is responsible for the tracking and the paging of UE in idle-mode. It is the termination point of the Non-Access Stratum (NAS)
- SGW is the point that interconnects the Radio Access Network (RAN) and the EPC. As its name indicates, this gateway serves the UE by routing the incoming

and outgoing IP packets. It is the anchor point for the intra-Long Term Evolution (LTE) mobility (i.e. in case of handover between eNodeBs) and between LTE and other 3GPP accesses.

- PGW is the point that interconnects the EPC and the external IP networks. These networks are called Packet Data Network (PDN). The PGW routes packets to and from the PDNs. The PGW also performs various functions such as IP address / IP prefix allocation or policy control and charging.

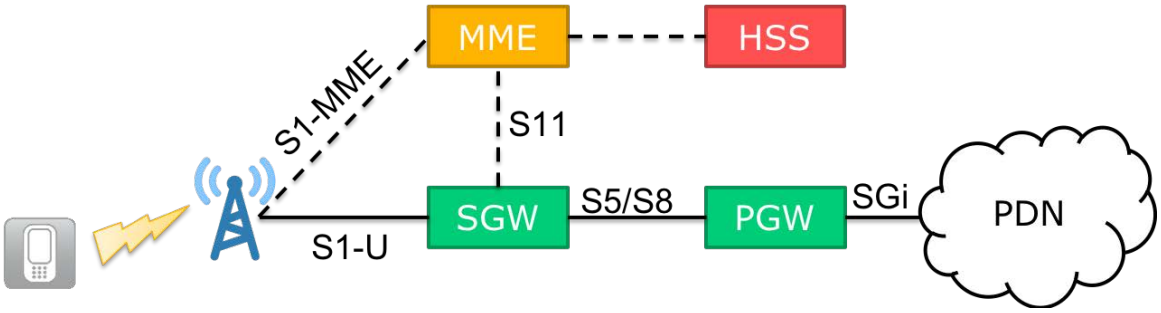


Figure 2.5: Evolved Packet Core Architecture

As introduced above, we can notice that some components of the EPC are dedicated only for the control plane while others are used in both the control and the data planes. The MME and the HSS for example are only control plane components. For those components only signaling traffic will be transiting on their control interfaces. On the other hand the SGW and the PGW are split into two parts: (i) the control part allowing to start a data session for a user as building the bearer, a tunnel, to be used to transit data, and (ii) the data part that consists on the default and/or the dedicated bearer for data traffic. As the control and the data plane are coupled in some of the EPC components, this architecture lacks of flexibility and elasticity. For example, if the data plane is congested and a scaling procedure is needed, the operator would deploy a new hardware component of the SGW and the PGW. In this case, as the control plane and the data plane are coupled in SGW and PGW, the operator will be scaling the control and the data planes while control plane scaling is not needed. In addition, those manipulations, costs a lot for the operators as new hardware components should be bought as also installation and configuration fee is added. To address these issues, key enablers listed in Section 2.3 allow to bypass some costs. It is in this respect that several core network improvements have been proposed, which are briefly described in the following sections.

2.4.2 Full Virtualisation

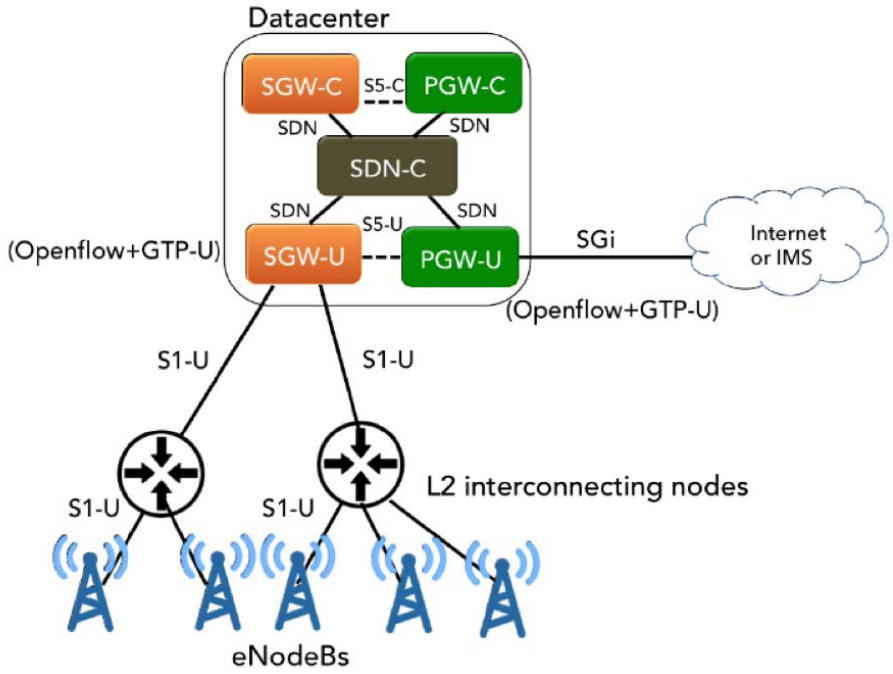


Figure 2.6: Fully virtualized EPC architecture

The full Virtualisation scenario consists on virtualizing the whole legacy architecture (i.e. EPC elements) using NFV, by hosting all the VNFs in a datacenter as illustrated in figure 2.6. In [31], [32] and [33], each of the SGW and the PGW are split into two types of entities: the U entities and the C entities. The U entities will be managed and configured by the SDN controller, while the C entities will act on top of the controller using REST API. Simple switches are used in the data plane to forward the traffic from the RAN or the Internet to the data center in order to be processed.

Also while virtualising and using NFV for mobile core network components, scaling is easier as it is a matter of software deployment or virtual ressource upgrade/down-grade. In addition with VNFs and SDN, reconfiguration or modification is easier as it is a software matter. Thus, this architecture adds scalability, flexibility and elasticity to the mobile core network.

However, additional latency may be observed as all the traffic is routed through the data center, a critical and a central point of the architecture. In addition, virtualising all the components of the core network and hosting them in virtual machines may also add latency. In order to tackle this issue, accelerators as eXpress Data Path (XDP) [34]

or Data Plane Development Kit (DPDK) [35] may be used. This aspect is out of scope of the thesis, interested readers may refer to the references listed above.

2.4.3 Control Plane Virtualisation

In this scenario, the SDN controller uses the control functions virtualised and running in a datacenter as REST APIs to push configuration into the modified switches implementing the S/P-GW-U functions. Compared to the first scenario, the users' data traffic in this case doesn't pass through a datacenter; it goes directly to the Internet using the mobile network operator gateway.

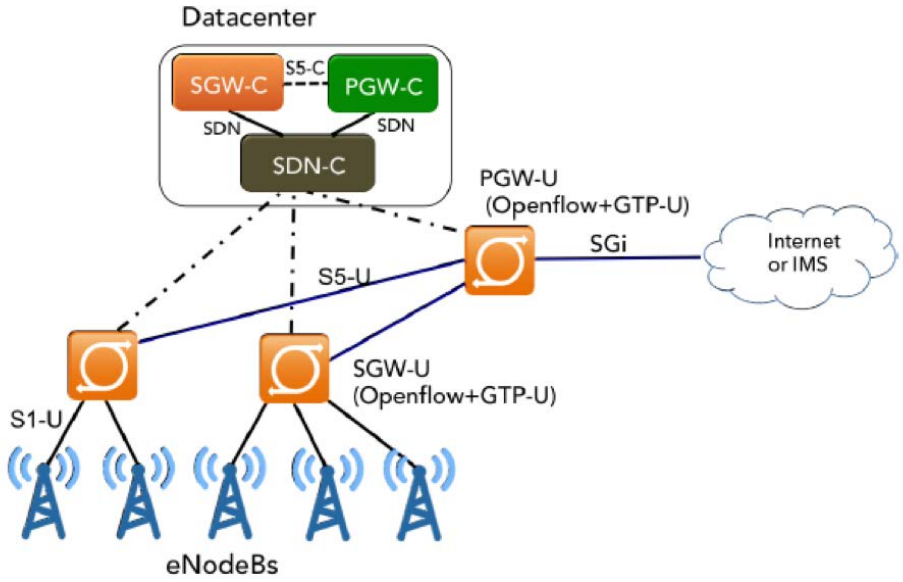


Figure 2.7: Control Plane Virtualisation EPC architecture

In this kind of architecture, different aspects were observed. Some authors proposed to use the OF protocol in the southbound interface and tag basis for routing. SoftCell [36], SoftMow [37] and [38] for example have taken this direction.

The SoftCell project uses the SDN architecture with a central controller for applying and pushing rules based on the application layer. Commodity switches are used in the core architecture for packet routing combined with middle boxes acting as transcoders, firewall or web caches. Finally access switches are used to classify packets sent from the UEs. The particularity of this project is to aggregate forwarding rules based on three tags: policy, location ID and UE ID. The advantages of this architecture are discussed in [38], where the authors present the benefits of the integration of SDN in the

EPC using tags based routing in different use cases. As SoftCell, SoftMow uses middle boxes in its architecture as transcoders, firewall or web caches next to programmable switches for packet routing. However, for the control plane, SoftMow adopts a hierarchical architecture. It decomposes the data plane into logical regions, each one controlled with a leaf controller. Each logical region is abstracted as a gigantic switch (GSwitch) and pushed to the higher level. The tag basis routing is explained by the global label pushed to each traffic group. On the other hand, the region-level controllers swap labels on ingress and egress switches.

In the same context, work in [39] proposes to employ tag based routing by using 802.1ad standard. As for the architecture, the authors propose to move the MME as a control API above the SDN controller. For S/PGW they propose either to keep them as they are, or to split them and move the control logic into the cloud.

Other works have proposed to use a different approach for integrating the SDN paradigm into the mobile core network. The solutions proposed in [40] and [41] use an SDN architecture in the mobile core network by applying flow based routing. These works propose to implement the control functions of SGW and PGW as applications on top of the controller and use OF switches for the data plane.

In [42], [43] and [44], the authors propose an SDN architecture with flow based routing. However, they moved the MME to the top of the SDN controller. So the controller contacts the MME to push flow rules in the switches.

In [45] and [46], the authors also propose to introduce SDN in the EPC by performing Flow based routing. They replaced the SGW and the PGW by Distributed Mobility Management Gateway (DMM-GW). The DMM-GWs are controlled using the OF protocol by the CROWD Local Controller (CLC). As SoftMow, CROWD adopted also a hierarchical architecture by deploying CROWD Regional Controller (CRC) on top of CLCs to manage them. In CROWD the authors decided to leverage mobility management to control applications. Also in [47] the same style of routing is proposed but this time with distributed SDN controllers.

Another way of routing is also proposed in the literature. It is based on GPRS Tunneling Protocol (GTP) routing. This method is more compliant with the 3GPP standard as it uses the same tunneling protocol. For example, work in [48] describes how to upgrade OF version 1.2 in order to support GTP protocol tunneling. In [49] the authors propose also an SDN architecture based on OF version 1.4, which introduces a new OF switch pipeline enabling virtual port for GTP/Generic Routing Encapsulation (GRE)

matching.

In the same spirit, works in [50], [51] and [52] propose SDN architectures in the mobile core network. They use GTP based routing in the data plane combined to OF protocol. For the control plane, they virtualise the MME, S/PGW control functions and run them on top of the SDN controller. Other works, like MobileFlow in [53], propose also GTP based routing using the Simple Multicast Forwarding (SMF) protocol [54]. The authors define two components: MobileFlow forwarding engines (MFFE) and MobileFlow controller (MFC). MFFEs are responsible for tunnel processing and charging, by using the OF protocol for transportation. The MFC is the SDN controller managing the network. The control plane and the data plane are also decoupled as the other architectures defined above. The particularity of MobileFlow is that the MFC will push GTP tunnels into MFFEs using the SMF protocol.

Following the same architecture scenario, control plane Virtualisation, authors in [55] propose a new routing aspect. They decided to use an IP based routing: Border Gateway Protocol (BGP). Moreover, they replaced the SGW by an EPC edge router (EPC-E). The latter is used in this context to terminate the GTP tunnel before entering the EPC. On the other side, a Core Router (CR) is defined. The control plane configures the CR using BGP protocol to update the table of the CR for routing in the data plane.

2.4.4 Hybrid / Mix Virtualisation

Finally the third and last scenario found in the literature is to use what is called a mix or hybrid architecture. In this architecture only the SGW or the PGW is decomposed and only C functions are virtualised. To understand more this architecture we illustrate in figure 2.8 the two aspects. In [56] and [57] the authors decided to split the SGW into SGW-C, for the control plane, and into SGW-U for the data plane. The SGW-C functions and the MME are virtualised on top of the SDN controller. The SGW-U implements the OF protocol and is configured by the SDN controller, while the PGW is kept as it is in the legacy EPC.

Authors in [58], propose to use the same scenario, but this time by decomposing the PGW and leaving the SGW as it is in legacy EPCs. PGW is firstly decomposed into PGW-C and PGW-U. The authors introduced a new component called PGW Orchestrator (PGW-O). PGW-O is responsible of managing the tunnels based on the instructions

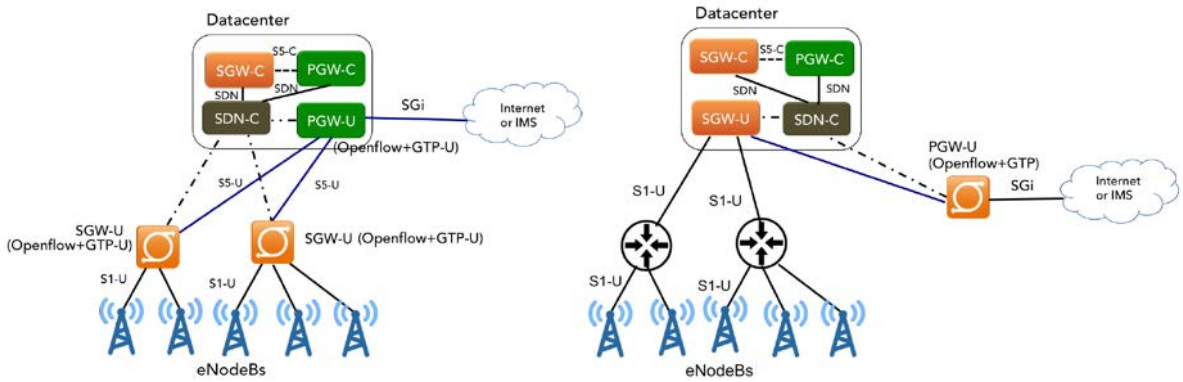


Figure 2.8: completely virtualised EPC architecture

of PGW-C. As for the PGW-U three new entities were introduced. The Packet Processing Unit (PPU) responsible for encapsulation/decapsulation of the GTP headers, the PGW-U Uplink switch (PUS) that dispatches the uplink traffic from the SGW to different PPUs and the PGW-U Downlink Switch (PDS) that routes the downlink traffic from external PDNs to certain PPUs.

2.4.5 Community research work summary

Figure 2.9 sum-up and classify the research work done on mobile core network architecture.

From the diagram we can notice that the community have split the research works based on three axes: (i) Traffic Routing Basis, (ii) Virtualisation scenario, and (iii) services. For the traffic routing basis research, we can cite IP based, Tag based, Flow based and GTP based routing. The virtalisation scenario are three: Full, Control plane, and hybrid Virtualisation. Finally, most of the service oriented research work are about mobility management and preserving/increasing the QoS.

Readers interested in more details and readings on core network architectures classification may refer to [59], [60] and [61].

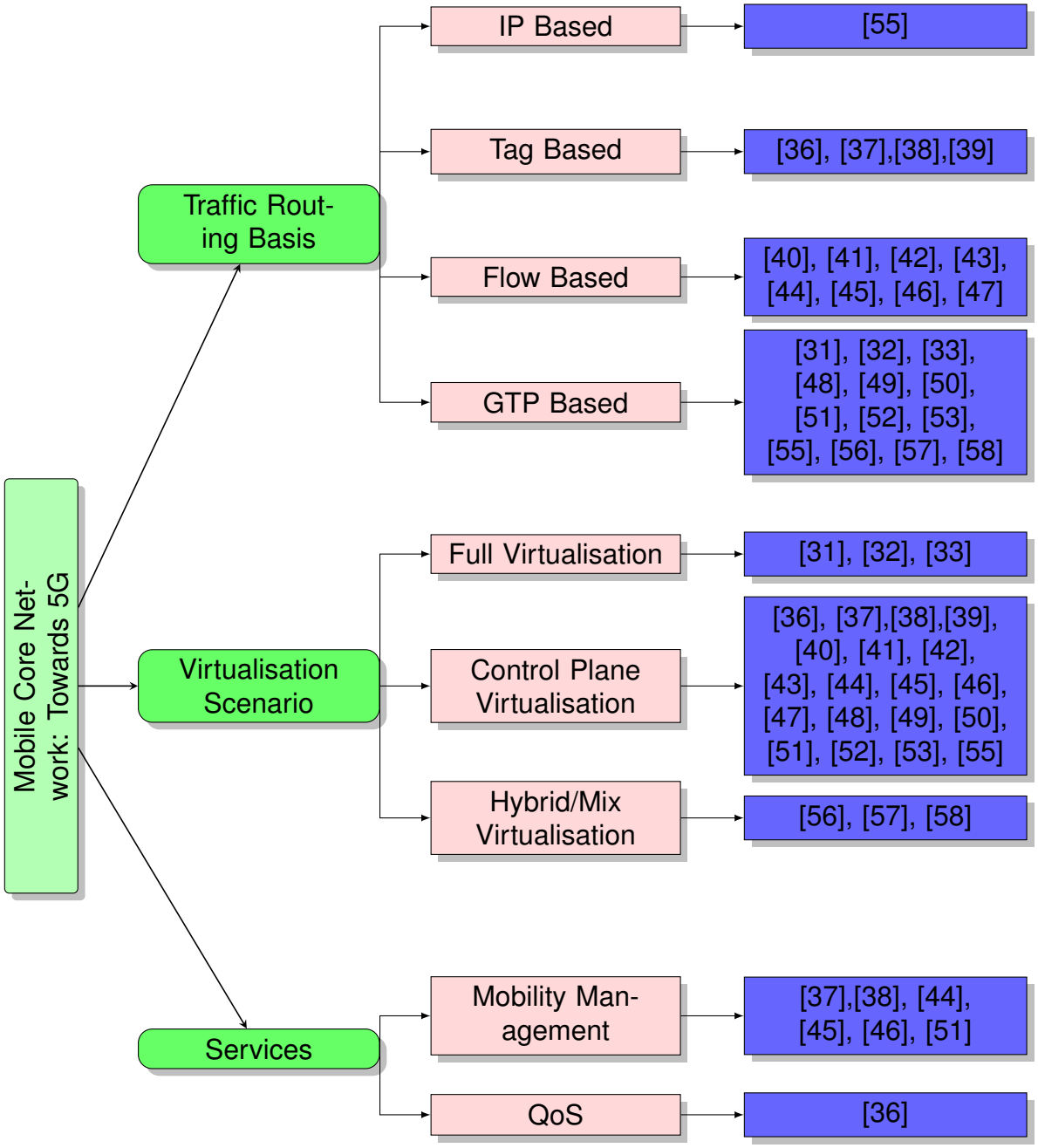


Figure 2.9: Mobile Core Network architectures: Towards 5G Classification

2.5 New Generation Core for 5G

Beyond the proposals and the research work of the community, the 3GPP group has started to define the new standard of the Next Generation Mobile core network.

The 5G core network architecture has been devised with the aims: (i) to keep compliance with the concept of 3GPP communication reference interfaces (point-to-point); (ii) to evolve in near future to a service-oriented architecture, where the 5G core control network functions communicate via a common bus, through the concept of producers and consumers. Indeed, 3GPP SA2 group has defined two architectures, one based on the reference interfaces [11], and the other one following a service oriented architecture. Regarding the 5G core functions, some are similar to 4G EPC one and some are completely newly designed. Notably, the access control and session management functions are combined in the MME of EPC, but separated in 5G core to better support fixed access and ensure scalability and flexibility. The main network functions, as defined in 5G Core, are:

- AMF, which handles access control and mobility. If fixed access is involved, the mobility management will not be required in the AMF. It is in charge of handling attach requests of UEs from several RAN; as being the entry point to the core network control plane it creates bottleneck issues when undersized;
- SMF, which sets up and handles sessions according to the network's policy;
- Policy Control Function (PCF), which corresponds to the Policy and Charging Rules Function (PCRF) in LTE. It is worth noting that this function will integrate also a policy framework for network slicing;
- User Plane Function (UPF), which can be deployed, based on the service type, in several configurations and locations to be able to handle and forward users' data traffic;
- UDM, which is similar to the HSS in EPS. However, it is envisioned that it integrates subscriber information for both fixed and mobile access in the NGC;
- NF Repository Function (NRF), which is a new feature. It provides registration and discovery functionality allowing the Core Network (CN) functions to discover each other and communicate via application programming interfaces (APIs).

Readers interested on having more details on these functions, and the other 5G core functions, may refer to [11].

In the following we shortly describe the AMF and SMF functions as this work focuses on the AMF scalability. Indeed, as shown in several studies [62], the AMF being

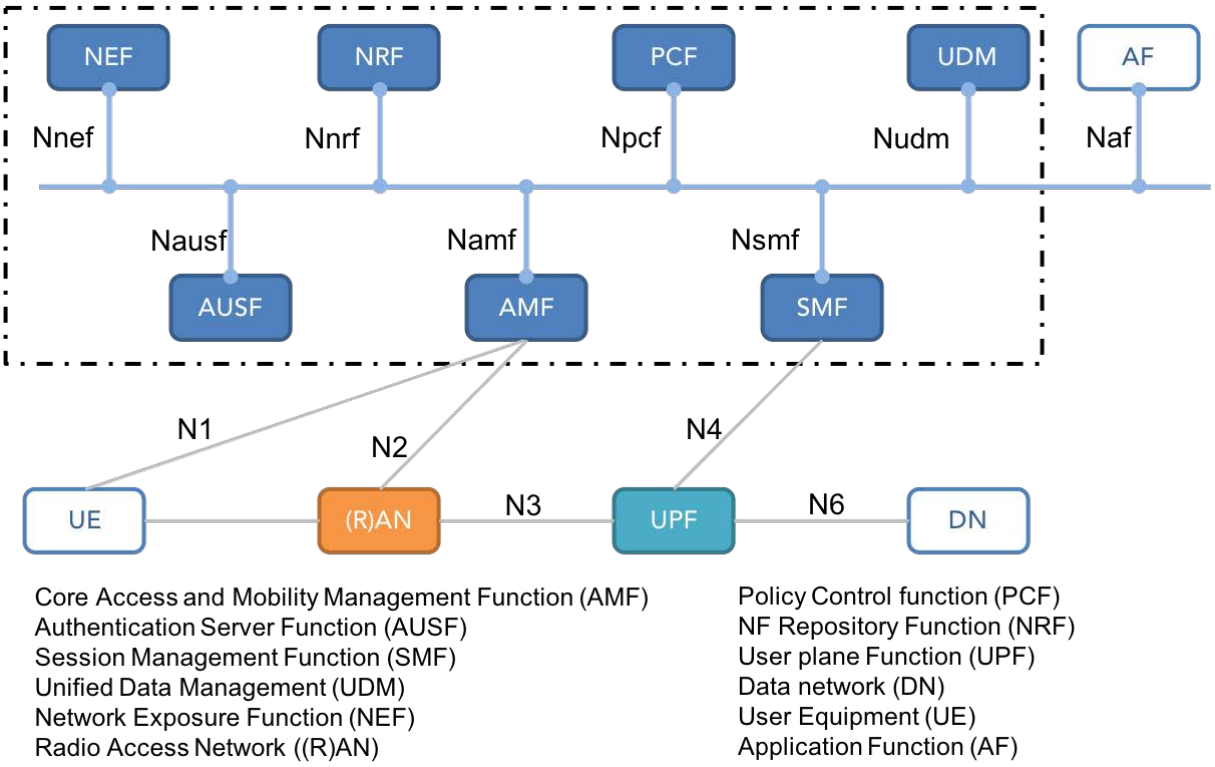


Figure 2.10: Service oriented 5G Core network

the entry point of the system, it is likely to suffer the most from the increase in arrivals, especially with the massive arrival of IoT terminals. For more details on the other functions reader may refer to [11]. The AMF ensures UE-based authentication, authorization, mobility management, etc. UEs, even using multiple access technologies, are connected to a single AMF, since the latter is agnostic to the access technology. The SMF is responsible for session management and IP address allocation to UEs. In addition, it selects and controls the UPF for data transfer. When a UE has multiple sessions, several SMFs may be allocated to each session aiming at managing the latter individually and possibly provide different functionalities per session. Based on operator policy or AF provided policy on packet flow, the PCF defines policies for mobility and session management, which will be enforced by the AMF and the SMF.

Figure 2.10 illustrates the service oriented 5G core network architecture. Only the UPF is deployed on the legacy transport network infrastructure; the other core network functions handling the control plane are expected to run in more centralized telco cloud platform. While the communications with the UE, RAN and UPF are using the reference interfaces, the core control plane functions are connected together via a common

communication bus. As stated earlier, the CN functions use the concept of producer/consumer, where a Core Network Function (CNF) may register, using the NRF, for specific events provided by another CNF via an API. Thus, either 1:1 or 1:N communication is possible. For instance, the AMF service exposes information to the other CNFs regarding events and statistics related to mobility; or a PCF provides all the operations related to policy rules to other CNFs. The service-oriented 5G core architecture could be easily deployed in a virtualised environment (e.g. VM or container), wherein libraries of functions may be requested from a VNF catalog and composed into virtual Core Network on demand.

2.6 Conclusion

In this chapter we have explained briefly the requirements and the expectations of 5G. It is expected to answer the need of massive connectivity demand, low latency, and high bandwidth. Also a short introduction of the key enablers of the 5G systems was listed: SDN for control plane and data plane separation, NFV for allowing to run the network functions on demand and in the cloud and finally network slicing for fulfilling the needs of multiple services with multiple requirements of QoS based on the same physical infrastructure. A review on the EPC and its main components, as well as the community work based on EPC toward NGC was explained. Thanks to this classification we have noticed that multiple Virtualisation scenarios were imagined and multiple techniques for traffic routing in mobile networking were targeted. Finally a short introduction of the NGC architecture and its components was listed.

Following this introduction of the general context, the next chapter presents the first contribution of the thesis. It consists on measuring the performances of three architectures relying on multiple KPIs.

CHAPTER 3

ON EVALUATING DIFFERENT TRENDS FOR VIRTUALISED AND SDN-READY MOBILE NETWORK

3.1 Introduction

Following chapter 2, SDN is one of the key enablers for evolving mobile network architecture towards 5G. SDN involves the separation of control and data plane functions, which leads in the context of 5G, to consider the separation of the control and data plane functions of the different gateways of the EPC, namely SGW and PGW. Indeed, the envisioned solutions propose to separate the S/P-GW into two entities: the S/P-GW-C, which integrates the control plane functions; and the S/P-GW-U that handles the UE data plane traffic. There are two major approaches to create and update user plane forwarding rules for such a partition: (i) considering an SDN controller for the S/P-GW-C (SDNEPC) or (ii) using a direct specific interface to control the S/P-GW-U (evEPC). In this chapter, we evaluate, using a testbed, those two visions against the classical vEPC, where all the elements of the EPC are virtualised. Besides evaluating the capacity of the vEPC to manage and scale to UE requests, we compare the performances of the solutions in terms of the time needed to create the user data plane. The obtained results allow drawing very interesting remarks, which may help to dimension the vEPC's components as well as to improve the S/P-GW-U management procedure. This chapter is organized as follows. First, a classification of the existing architectures is provided and it lists the major differences between them. Then the deployed architectures for the benchmark in addition to the scenarios and the test methodology are presented. Finally, results are analyzed and discussed.

3.2 Related work

A high number of research works have been conducted to discuss and propose solutions to virtualise the EPC and enable the "softwarization" of the mobile core network, featuring SDN and NFV, as a first step towards 5G. As mentioned earlier, two major visions exist. The clean slate architecture unveils a complete novel and flat architecture composed by OF switches and SDN controllers. As an example, the solution presented in [44] withdraws GTP tunnels and proposes a full IP flat architecture managed by SDN controllers hosted close to the MME, while the S/P-GW are replaced by common OF switches. Although the proposed solution is novel, it contains many concerns, particularly the lack of scalability and QoS support.

The second vision relies on the current 4G architecture, keeping compatibility with the 3GPP specifications, to include SDN into the mobile core network. Most of the existing works, including 3GPP CUPS initiative, belong to this category and mainly assume a separation of data and control plane functions. The difference between these works consists in using or not virtual instances of the EPC entities (i.e. MME and S/P-GW), and specifying which of them should be split.

In addition to the idea of separating the control and data plane functions, the authors in [31] distinguish resource management and signaling functions at the control plane level. Accordingly, the authors proposed four architectures: (i) Full cloud architecture, whereby all EPC entities (control and data planes) run as VNFs hosted in the cloud. Although this architecture is entirely virtual, it does not use SDN; (ii) Control-plane migration proposes hosting all control plane functions and elements (i.e. MME and S/P-GW-C) in the cloud, while keeping data-plane functions running on dedicated physical machines; (iii) Signaling-plane architecture, wherein the MME and control signaling functions of the S/P-GW (i.e. tunnel establishment, etc.) are hosted in the cloud, while data plane and resource management functions remain outside the cloud, running on dedicated machines; (iv) Scenario-based architecture follows the same principle as the signaling-plane architecture. The difference concerns the location where the signaling control is executed. Indeed, this architecture distinguishes between delay-critical scenarios, wherein the signaling control should run on the data plane, and CPU-intensive scenarios, which run signaling control in the cloud.

In [63] the authors propose to split the SGW into SGW-C, for the control plane, and SGW-U for the data plane. The SGW-C as well as the MME are virtualised on top of

an SDN controller. The SGW-U implements the OF protocol and is configured by the SDN controller, while the PGW was left unchanged.

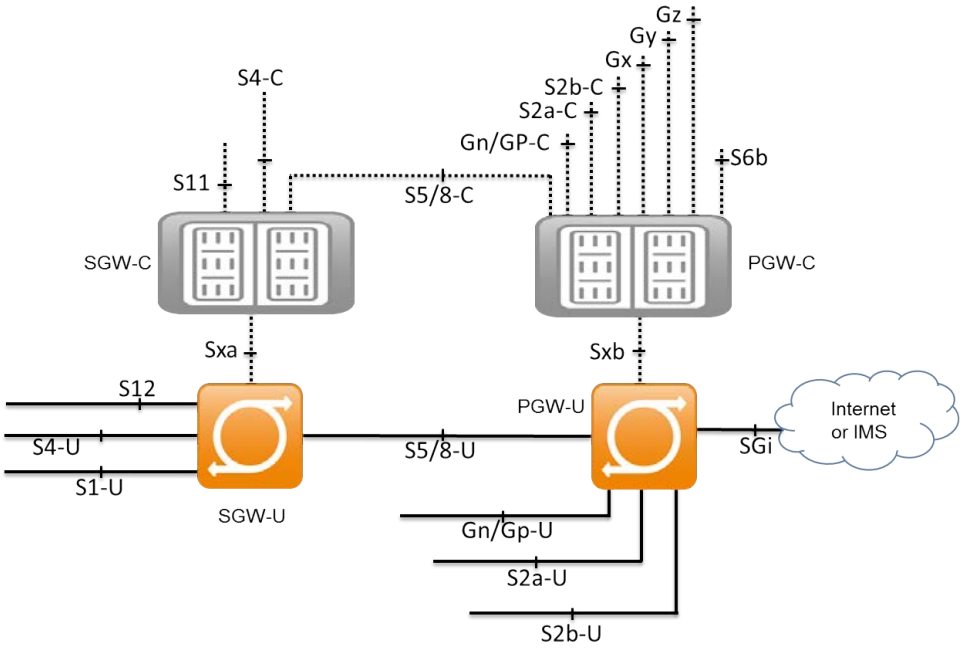


Figure 3.1: 3GPP TR 23.714 Split Architecture.

Relying on the function separation of the S/P-GW, the authors in [51] propose to reorganize the EPC by creating a new element, namely Mobile Controller (MC), which combines the MME and the S/P-GW-C functions. In addition, the S-GW-U and P-GW-U are merged into one element, namely GW-U, which is remotely controlled by the MC via the OF protocol. Due to the topic relevance and to prepare the evolution towards 5G, the 3GPP, via the CUPS group, is developing a set of specifications and new interfaces in order to separate the data and control planes functions. The new elements and interfaces that merged from the function separation process is depicted in Figure 3.1. The main difference, with the above works based on data and control plane separation, is about not using SDN controllers. Indeed, the CUPS architecture splits the SGW and the PGW, while defining new interfaces: Sxa interface connecting the SGW-C to SGW-U and the Sxb interface connecting the PGW-C to the PGW-U. In addition, referring to Figure 3.1, we can notice that the interfaces S11 and S1U are kept to connect, respectively, the MME to the new SGW-C and the eNodeB to the SGW-U; while interface S5/8, which was used to connect the SGW to the PGW in the legacy EPC, is split into S5/8 C and S5/8 U to connect, respectively, the SGW-C to the P-GW-

C and the S-GW-U to the P-GW-U. Last but not least, the CUPS neither introduced an SDN controller in their architecture, nor interfaces preparing its possible integration in the future. Thus, we can assume that the CUPS preferred to only adopt the control and the user planes separation, without even a reference to OF. Basically, the CUPS group is intending to use the GTP protocol with extensions or upgrades.

3.3 Benchmark Architectures

Having reviewed the different initiatives, from industry and academia, to integrate SDN in mobile networks, we present in this section the deployed architectures for our benchmark.

3.3.1 Virtual EPC (vEPC)

The first selected architecture (vEPC) as mentioned above represents the most classical one. The HSS, the MME, and the S/P-GW are each deployed on a separate VM and interconnected using the 3GPP reference interfaces (i.e. S1 and S11). The data plane is also following the 3GPP specifications using GTP bearers with the GTPv2 protocol. This architecture is based on OpenAir-CN project [8].

3.3.2 SDN EPC

The second selected architecture (SDNEPC) is compatible with the 3GPP specifications, and aims to evolve the current 4G architecture towards integrating SDN. We omitted considering the “clean slate” solution (introduced in Section 3.2), as the latter is unlikely to be adopted due to the weaknesses mentioned in section 3.2. Thus, this architecture integrates an SDN controller based on OpenDayLight (ODL) [9] with a REST API that we developed to support 3GPP procedures. Figure 3.2 depicts the mobile network architecture featuring SDNEPC. Hence, it splits S/P-GW-C from S/P-GW-U. The S/P-GW-C is deployed on a VM and communicates with the ODL controller via a REST API through HTTP. Our programmable user plane relies on a modified version of OVS [10] that we have patched (mainly to match GTP header) in order to perform as an S/P-GW-U. It provides an endpoint for the GTP tunnel created with the eNodeB. From there the traffic is routed using OF protocol, according to the flow rules pushed by ODL,

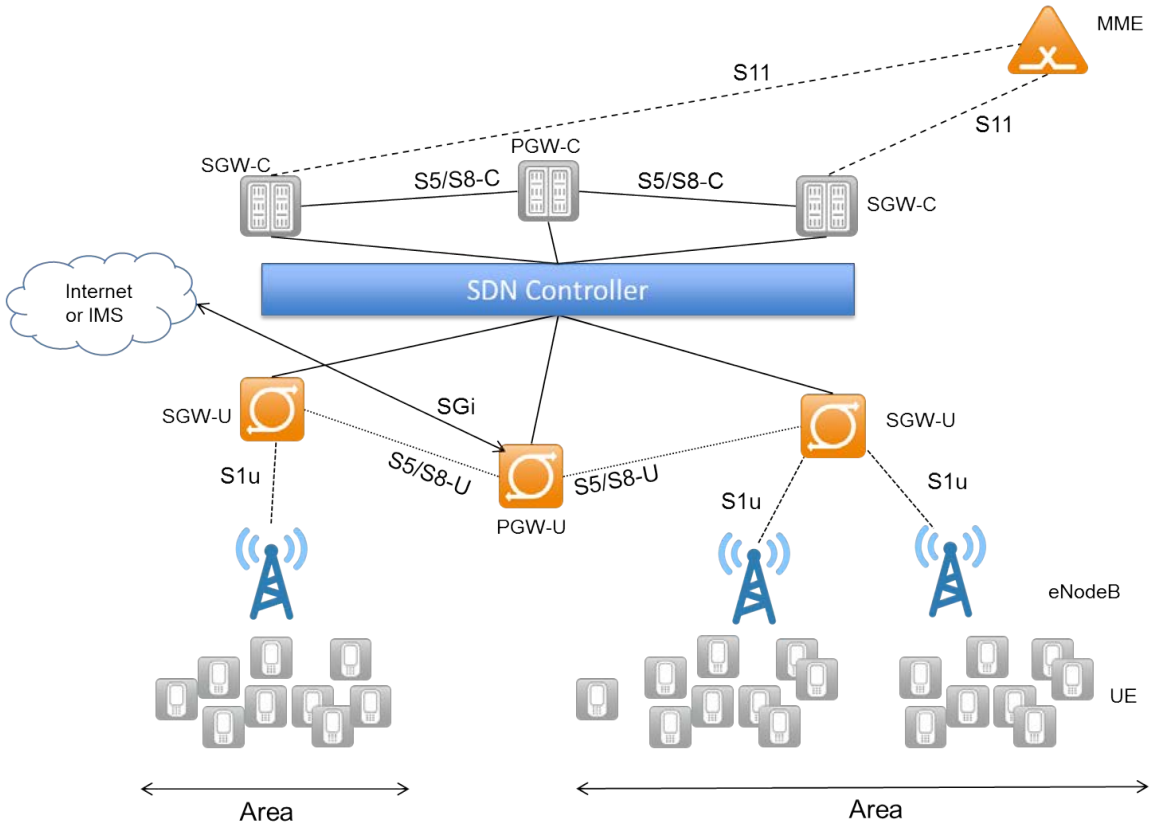


Figure 3.2: EPC with SDN integration.

following the request from the S/P-GW-C entity. In Figure 3.2, each SGW-U is connected to several eNBs using the S1-U interface. The interface between the SGW-C and the SDN controller is based on REST API. The MME is connected to the SGW-C via the S11 interface, and we assume that the SGW-C is connected to the PGW-C via S5/S8-C (specified in CUPS) interface. Note that one SGW-C (resp. PGW-C) may manage one or more SGW-U (resp. PGW-U). For the sake of simplicity, our implementation merges the SGW and PGW in one element, and the latter is split into S/P-GW-C and S/P-GW-U to integrate SDN. Figure 3.3 illustrates a simplified example of creating a default bearer in case of using an SDN controller. All the messages in black are 3GPP standardized messages. The ones in blue are SDN specific messages. After the RACH procedure, the UE begins by requesting a Radio Resource Control (RRC) connection with the eNodeB. The latter triggers the creation of a NAS session for the connected UE. After checking the UE authorization, the MME triggers a request to the S/P-GW-C to create a session and to initiate the creation of a default bearer for the UE.

The S/P-GW-C initializes the session context for the UE, and creates Bearer (i.e. tunnel ID), which will be communicated to the MME. The latter communicates the Tunnel ID and other information to the eNodeB, which in turn initializes a Tunnel ID and communicates the value to the S/P-GW-C via the MME. Once the eNodeB's tunnel ID is received, the S/P-GW-C sends an SDN UE Setup rule to the SDN controller in order to enforce the bearer at the concerned S/P-GW-U. The Setup rule is implemented using the REST API, which includes the tunnel IDs and the end-point tunnel IP addresses (i.e. the IP addresses of the eNodeB and the S/P-GW-U). The following is an example of a JSON configuration sent from the S/P-GW-C to the REST API:

```
{"input": {
        "endpoint-id": "3",
    "remote-ip-address": "10.10.25.6",
    "remote-teid": "4058422",
    "device-id": "ovs:1",
    "local-teid": "9",
    "ue-ip-address": "192.168.0.2"
}}.
```

The SDN controller translates the REST API message into OF rules to be sent to the S/P-GW-U. Two rules should be enforced, for the Uplink and the Downlink. The detach

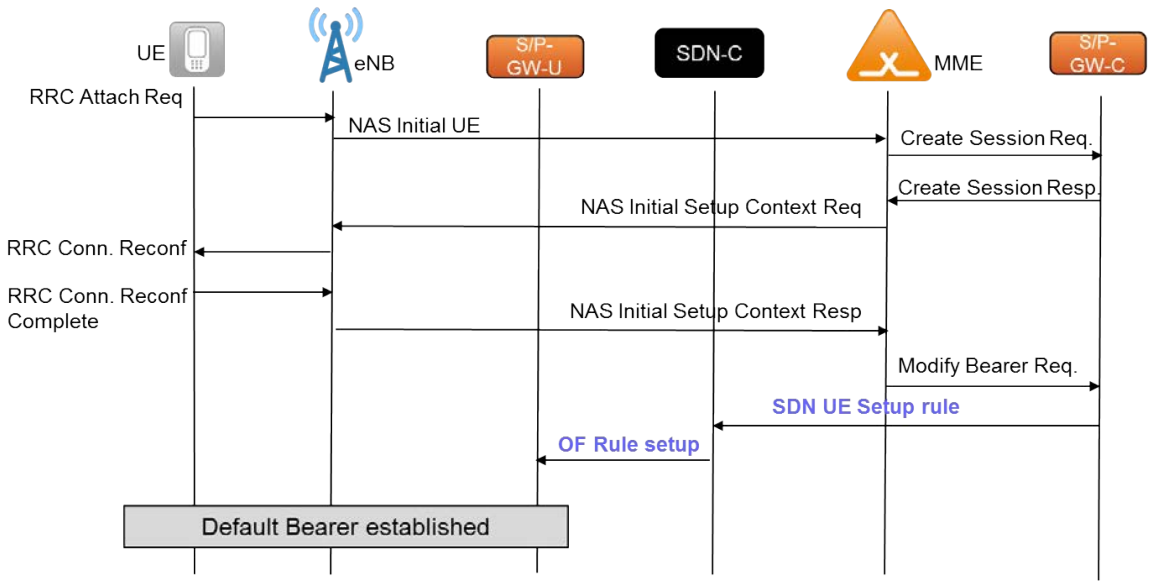


Figure 3.3: UE Attach Call Flow with SDN integration.

procedure and the dedicated bearer creation procedure share the same spirit, i.e. the data plane traffic is separately managed by the SDN elements and controlled by the S/P-GW-C via the SDN controller.

3.3.3 Enhanced EPC

Finally, the last selected architecture (evEPC) is merely similar to the previous one, except that no SDN controller is used. The S/P-GW-C directly pushes the flow rules using OVS commands via a direct interface. Through this architecture, we are able to mimic the 3GPP CUPS proposed architecture.

3.4 Test scenarios and methodology

To emulate the eNodeB and UEs, we used the emulation mode of OAI [7], known as OAISIM. Only physical channel is simulated. Protocol standard stacks from the Medium Access Control (MAC) to the Packet Data Convergence Protocol (PDCP) layers are following operations as real eNodeB. However, the UE part of OAISIM wasn't able to launch more than 1 UE. Thus we have developed the needed parts in order to enable the deployment of several UEs in parallel on the same machine (i.e. several instances of NAS layer). Therefore, we are able to generate (up to 900) simultaneous UE connection requests (NAS and RRC). It is worth mentioning that this multi-UE feature is pushed to the open-source community of OAISIM.

1st Conf	CPU	RAM
HSS	1	1024
MME	1	1024
SPGW	1	1024

2nd Conf	CPU	RAM
HSS	1	1512
MME	2	2048
SPGW	2	2048

Table 3.1: Configuration template of the EPC modules.

Since our first objective is to evaluate the performances of the virtualised instances of the EPC, we selected two different VM configurations ; consisting in changing the amount of CPU and RAM dedicated to VMs running the MME and S/PGW. Table 3.1 details the amount of CPU and RAM allocated to a VM. The first configuration dedicates one CPU and 1 GB of RAM to the VMs hosting, separately, the HSS, MME and S/P-GW; while the second one doubles the resources dedicated to the VM, except for the

HSS. Indeed, thanks to rehearsal tests, it was observed that the HSS element does not consume much CPU or RAM. Therefore, the VM running the HSS element will keep its 1 CPU for both configurations; but its RAM will be slightly increased in the second configuration. On the other hand, the CPU and the RAM of the VMs running the MME as well as the S/P-GW will be doubled, aiming at clearly demonstrating the effects of increasing/decreasing the CPU or the RAM on the performances. The S/P-GW-C, for the second and third architectures, uses the same configuration as the S/P-GW.

KPI	Definition
Control Plane Attach Duration	Elapsed time between an attach request and an attach accept
Control + Data Plane Attach Duration	Complete control and data plane attach duration
Processed UE requests during one second	Successfully attached UEs comparing to received requests during one second
CPU Consumption	VM CPU Consumption

Table 3.2: Testbed KPIs

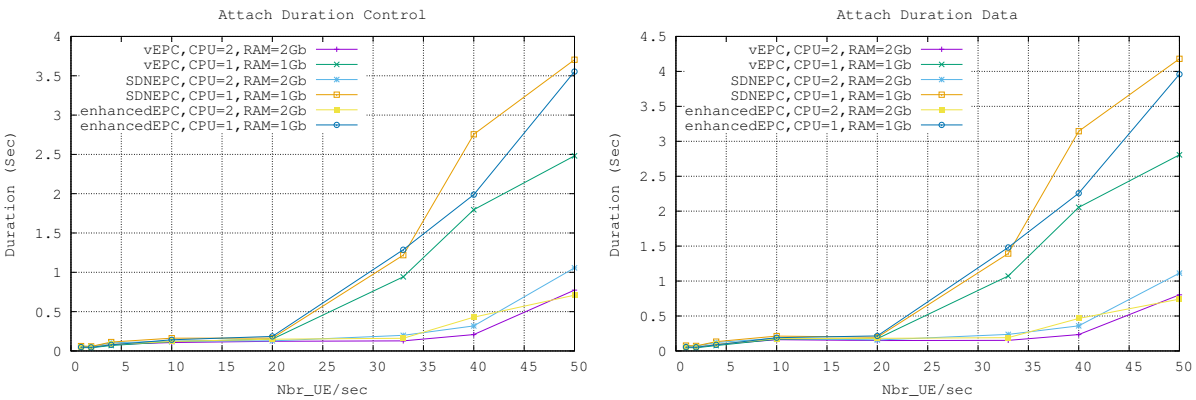
All the emulated UEs connect to the mobile network via the same emulated eNodeB. With this number of UEs (900 UEs), combined with the technical tests listed below, we are able to clearly show the vEPC elements' performances and their capacity to manage UE requests. Each test consists in launching a predefined number of UEs per second, until reaching the maximal number of emulated UEs. Launching a UE consists in sending the RRC messages, followed by the NAS attach initialization. The realized scenario increases the number of attach per second from 1 UE to 50 UEs per second. The tests are stopped when reaching the maximum number of UE, i.e. 900 UEs. For each generated point, i.e. a number of UE per second and a specific vEPC configuration, the test is repeated 30 times to obtain accurate results.

Since our objective is to evaluate the performance of the three architectures: vEPC, enhancedEPC, and SDNEPC; we defined four KPIs as listed in Table 3.2. The first one is the control plane attach duration, which represents the elapsed time between an attach request and an attach accept referring to the 3GPP standard. In other words, it is the time taken by the MME (and HSS) to identify, authenticate and register a UE in

the network. The second KPI is the complete control and data plane attach duration. It is the time elapsed between an attach request and the modify bearer response of the S/P-GW in the vEPC architecture. For both enhancedEPC and SDNEPC versions, it is the time elapsed between an attach request and the moment that OVS creates the two OF rules in regard to UE data plane traffic (uplink and downlink). The third KPI indicates the ratio of treated UE requests during one second; i.e. the ratio of the number of UEs successfully attached by report to the number of attach requests received during one second. The last KPI gives the CPU consumption of the vEPC entities (mainly the VM hosting the MME). The first, third and fourth KPI are used to investigate the capacity of the vEPC to scale with the number of requests, and the response time of the vEPC to completely attach a UE. Besides indicating the capacity of the vEPC to establish the UE data plane, the second KPI allows to compare the performance of the three architectures. Indeed, one important concern of splitting the S/P-GW and integrating an SDN controller is the added latency to establish the UE data plane.

3.5 Performance evaluation

3.5.1 Results analysis



(a) (b)

Figure 3.4: Attach duration over (a) control plane, (b) control and data plane.

Figure 3.4a illustrates the average attach duration of 900 UEs, over the control plane, by report to the number of attach request per second. This figure includes the

performances of the three architectures featuring the two configurations. Based on the VM configuration, the performance increases proportionally to the CPU and RAM amounts. For the first configuration, i.e. only 1 CPU and 1 Gb of RAM, the vEPC handles very well the attachment procedures until 20 attach requests per second. Beyond this number, the average of the attach duration starts to increase. In addition, we notice that the second configuration pushes the limits of the vEPC to 40 attach requests per second. By comparing both configurations' results, we are able to conclude that doubling the CPU and the RAM size allows doubling its capacity to process more users' requests. Also, we remark that in both configurations, the three selected architectures have nearly the same performances. This can be explained by the fact that the three architectures involve the same control plane entities (i.e. MME and HSS) for the attachment procedure. The difference involves only the data path construction, while the MME and the HSS modules remain the same for the three architectures; i.e. the functions related to the attachment procedure.

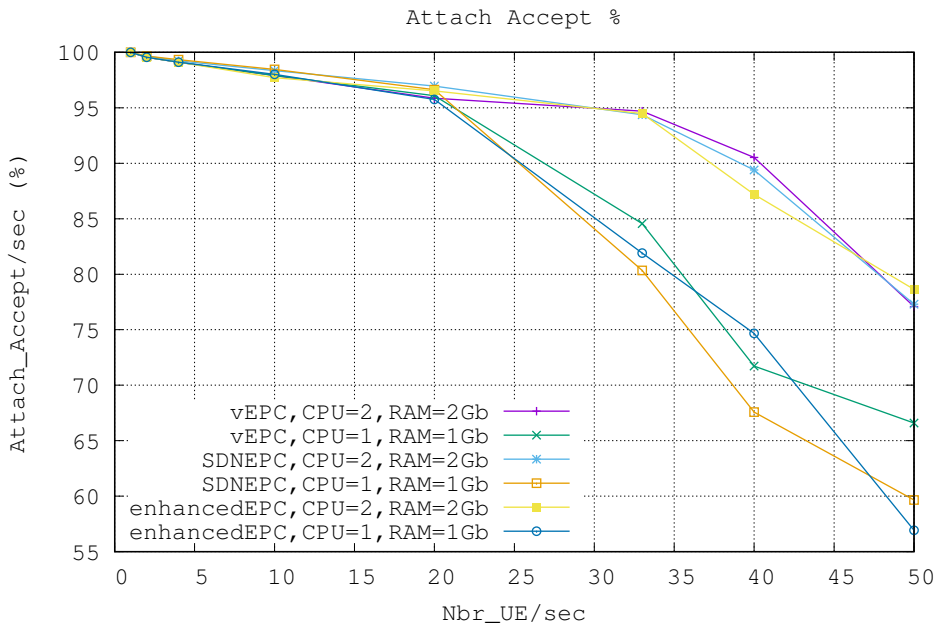


Figure 3.5: UE accepted ratio over the C plane.

Figure 3.5 indicates the ratio of accepted UE by report to the number of requests over the control plane. As in the previous figure, we observe that the three architectures have a similar performance. For both configurations, we remark the same thresholds, as in Figure 3.4a, from where the vEPC is not able to follow the pace of UE attach

requests. Indeed, when the number of requests increases up to 20 (resp. 40) for the first (resp. second) configuration, the average attach duration increases indicating that some requests were not satisfied during the considered time interval of $1s$.

Figure 3.4b illustrates the average duration of the attach request over both the user and control planes versus the number of request per second. The figure includes the three architectures using the two configurations. First, we observe that, for the three architectures, the attach duration over the data plane is almost the same as the attach duration over the control plane; which means that the datapath is built on demand with insignificant latency.

Second, we remark that the performance of the vEPC, the enhancedEPC and the SDNEPC version is slightly different. When the EPC is not congested; 20 UEs per second for the first configuration and 40 UEs per second for the second one; the difference between the vEPC and the enhancedEPC is about $1ms$. This is mainly due to the half of an RTT (one way communication) between S/P-GW-C and OVS (the Round-Trip Time (RTT) was around $2ms$). For the SDNEPC, the attach duration average is merely $2ms$ more than the vEPC. We argue this value by the RTTs of connections between the S/P-GW-C and the SDN controller, and between the SDN controller and S/P-GW-U.

	1 UE/sec	2 UE/sec	4 UE/sec	10 UE/sec	20 UE/sec	33 UE/sec	40 UE/sec	50 UE/sec
1st Conf	7%	12.63%	22.22%	44.22%	96.52%	98%	100%	100%
2nd Conf	3.53%	7.02%	11.65%	30.48%	51.76%	77.78%	79.8%	80.76

Table 3.3: MME CPU usage over pattern tests.

In order to check the root reason of the performance degradation (as shown Figure 3.4a and 3.5), we draw in Table 3.3 the CPU usage of the VM (hosting the MME) for both configurations, focusing only on the case of vEPC. Indeed, as illustrated in Figures 3.4a and 3.5, the architectures merely achieve the same performances. From this table, we clearly observe that the points from where the attach duration are exponentially growing, i.e. 20 and 40 for configuration 1 and configuration 2, respectively, correspond to high CPU usage. Accordingly, we were able to saturate the VM capacity, leading to degrade the performance of the MME. This observation is very important for dimensioning the VM hosting the MME element; we have the choice to double the VM capacity or to deploy multiple instances of the MME.

3.5.2 Discussion

From the results illustrated in Figure 3.4b, we clearly conclude that using an SDN controller doesn't introduce a consequent delay to establish the data plane for UEs. It is worth mentioning that integrating the SDN paradigm in mobile networks gives more flexibility for introducing new services especially with the programmability offered by NFV. Also, decoupling the control plane and the data plane ensures flexibility in system scalability. Thus, the control plane can be scaled without scaling the OVS function (data plane) and vice versa unlike classical S/P-GW which needs to be scaled completely even if one of the planes is not over-headed. Although this argues in favor of using SDN and NFV in 5G, still, there is a need for optimization techniques because we can clearly conclude that the OF protocol is not adapted for the mobile network use-case. Indeed, in traditional IP network, an OF rule may be used to forward the traffic for many services and many users at the same time. Unfortunately, in mobile networks, two flow rules are needed for each service on a UE. For example, to configure the data path of one service (i.e. one GTP bearer) used on the UE, it needs the installation of two flow rules: one flow rule for the uplink and one flow rule for the downlink. Each bearer needed by a user will obligatory lead to the installation of flow rules. This factor affects data plane by adding more latency in OVS, due to processing a huge table of flow rules, each time a packet needs to be routed. Finally, the obtained results open several possibilities to improve the deployment of virtualised EPC. The function depicted in Figure 3.4a allows to better dimension the vEPC elements, particularly the MME. Indeed, for both configurations, there is a point from where the attach duration grows exponentially, showing a clear degradation of the response time of the MME; which is mainly the results of the VM CPU saturation. Solutions could consist in scaling the system by either increasing the VM capacity or adding an additional MME VM as well as a Load Balancer between the different MME instances. Clearly, the choice between these two solutions has to further be investigated.

Additionally, the results depicted in Figure 3.4a could be used to dimension the vEPC before the deployment. To do that, we fit a polynomial function to our measurement results (Figures 3.4a). We found that the following expression reasonably describes our data (the coefficient of determination in our fit was $R2 = 0.9992$) and approximates the average attach duration by report to the number of UE requests per second as follows: $f(x) = 0.07 + 0.026x - 4.27x^4 + 3.96x^5$. Where $f(x)$ is the average attach duration and x the number of requests per second. Thus, it is possible to predict

the VM size according to a target average attach duration.

3.6 Conclusion

In this chapter, we used a testbed platform (based on open source tools) to evaluate the performances of: (i) a vEPC; (ii) an evEPC where data and control plane are decoupled; (iii) an SDNEPC where an SDN controller is deployed. We were interested on metrics regarding the control plane and data plane attach performances. The obtained results allow us to draw conclusions towards dimensioning the virtualised version of EPC, in terms of CPU capacity, as well as the flexibility and the scalability brought by SDN and NFV in mobile networks.

However the tests results of this chapter have shown a performance problem concerning the MME. As the MME is a single point of access to the core network, it may get overloaded easily as the number of connected devices is tending to increase in 5G. Obviously, there is a need to have a mechanism to solve this issue, which may consist on either: (i) adding more resources (CPU and memory) to the VM, i.e. scaling-up; or (ii) adding a new VM to the MME (with an internal load balancer), i.e. scaling-out.

Therefore in the next chapter we will propose a Congestion Control mechanism in order to remedy to this issue.

CHAPTER 4

CONGESTION CONTROL IN NEW GENERATION CORE NETWORK OF 5G

4.1 Introduction

One of the requirement of 5G is to support massive number of connected devices, considering many use-cases such as IoT and massive MTC. While this represents an interesting opportunity for operators to grow their business, it will need new mechanisms to scale and manage the envisioned high number of devices and their generated traffic. Particularly, the signaling traffic, which will overload the 5G core control NF in charge of authentication and mobility, namely AMF. The objective of this chapter is to provide an algorithm based on Control Theory allowing: (i) to equilibrate the load on the AMF instances in order to maintain an optimal response time with limited computing latency; (ii) to scale out or scale in the AMF instance (using NFV techniques) depending on the network load to save energy and avoid wasting resources. Obtained results via computer simulations indicate the superiority of our algorithm in ensuring fair load balancing while scaling dynamically with the traffic load. This chapter is organized as follows. First a related work on legacy MME scaling and legacy MME load balancing in SDN and NVF paradigms is presented. Then the assumed architecture is presented based on NGC and NFV, and the interaction of the control model with the assumed architecture is explained. This is followed by the description of the proposed analytical model for AMF load balancing and scaling. Finally, model evaluation and results analysis are presented.

4.2 Related Work

In legacy 4G EPC, Mobile Network Operators (MNO) developed different strategies to load balance the MMEs in a way to avoid network congestion. Some, for example, deploy multiple MMEs for a granted zone, and use probabilistic distribution of UE arrivals at the eNodeB level. Others may just deploy one over dimensioned MME for a given zone, and thus no load balancing procedure is needed at the eNodeB. However, experiences show that even an over-dimensioned MME is subject to persistent overloads [64]. Besides load balancing, dynamic scaling, as offered by NFV, is not really possible in 4G due to the difficulties to instantiate new MMEs on demand.

Nevertheless, in 5G, modern opportunities emerge to tackle this issue in an economic and intelligent manner. Indeed, with SDN and NFV the network control functions of the MME are virtualised and hosted in the cloud. Thus, the deployment of a new MME will be a matter of software deployment. This modern way of dimensioning the network is cheaper and faster than the classical solution that is based on hardware. In fact, many studies and reflections were conducted on the adaptation of SDN and NFV for 5G networks, i.e. in [65] and [66]. In [67] and [68], authors propose analytical models providing a quick way to help mobile operators to plan and design network optimization strategies without large-scale deployment, saving on cost and time. In [69], authors present an NFV-based traffic offloading framework architecture using vEPC, aiming at enabling on-demand traffic offload when the legacy EPC network capacity is reaching an offload threshold. This can be considered as a transition solution from 4G to 5G.

Some other works have been conducted addressing the MME scaling and load balancing issues within the context of 5G. Authors in [70] propose a new model of state-full MME. They propose to split the MME into three parts: 1) the traffic sorter, 2) the processing functions and 3) the state database. This approach proposes creating groups of International Mobile Subscriber Identities (IMSI) based on hash value and assigning a processing function instance to a group of UEs. Based on the IMSI group, the traffic sorter will route UE requests to specific processing function. However, this limits the scalability of the processing functions, as there should always be at least one worker available to serve each group of UEs. Further, this work proposes only horizontal scaling (processing functions scaling), which is limited by the processing capability of the traffic sorter, as it is the single point of access to the MME node;

hence, not addressing MME node scalability.

A distributed MME model is also proposed in [71] and [72]. In contrast with the approach proposed in [70], the MME model is stateless and based on an external users' state storage system. Thus, the migration between MMEs is limited only to UEs in an idle state. However, in the active state, UEs are attached to an MME instance. Therefore, another MME instance may receive a network event for that UE. This request has to be forwarded to the correct MME, hence increasing latency of EPC procedures.

Authors in [73] and [74] propose a stateless vMME that is split into three logical components: 1) front-end (FE), 2) MME service logic/Worker (SL), and 3) state database (SDB). The authors assume that MME SL can handle any request for any UE as they are stateless. However, as the users' state database is local and is not shared with others vMMEs, this limits requests' handling within the same MME. The authors, also, propose to use one FE element, which may represent a congestion point.

4.3 Architecture

In this work, we assume a complete New-Generation Core (NGC) based upon the 3GPP work [11]. The overall architecture is depicted in Figure 4.1, which illustrates the new core Network Functions (NFs), where some of them are the result of a split of the current EPC functions, such as the MME. The communication between those functions will be held by reference interfaces, noted as Nx as depicted in Figure 4.1. In this chapter, we will direct our focus only on the Core Access and Mobility Management Function (AMF), the Session Management Function (SMF) and the Unified Data Management (UDM) as they are approximately the equivalent to the MME component in LTE. For further information concerning the other functions, please refer to [11]. In the NGC architecture, the legacy MME component is split into three NFs: the AMF, the SMF and the UDM. The AMF handles only access control and mobility requests, the SMF will manage the UE sessions and the UDM will handle the UE contexts. Indeed, this approach offers more flexibility to the core network, but still the AMF may suffer from signaling traffic overload. Therefore, we propose, complimentary to the 3GPP work, a mechanism that addresses the AMF scalability by providing a novel scale in/out algorithm depending on the network load. Via this approach, a single UE procedure is processed by a unique AMF. The other procedures may use, however, a different AMF as the UEs states and the sessions are stored respectively in the UDM and the SMF.

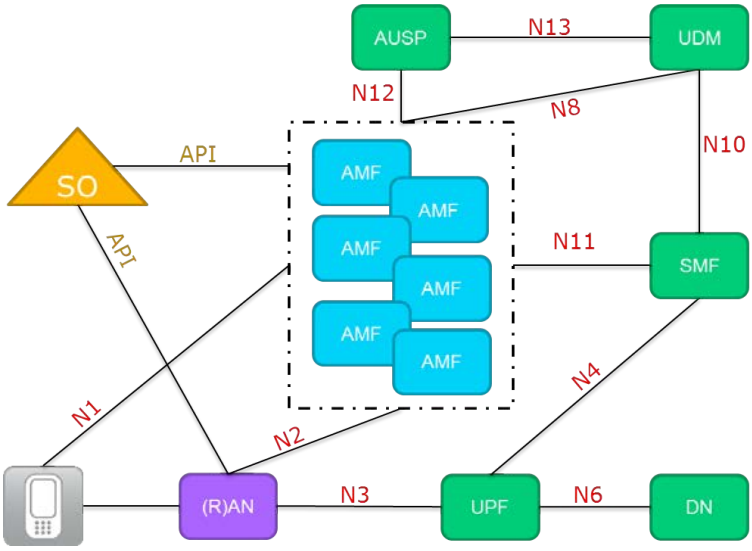


Figure 4.1: Architecture proposal for 5G

To scale in/out the AMF, a Service Orchestrator (SO) is needed. It will be in charge of running the proposed control model, described below, and deploying new AMF instances when needed. The SO could take part of the NFVO, as defined by ETSI NFV model [24]. The SO will use management interfaces (or API) to communicate with the AMF and the RAN functions (ex. eNodeB). Moreover, as specified by the ETSI NFV model, the SO should be connected to a VIM in order to add or remove virtual AMF instances, and to the Network Functions Virtualisation Manager (NFVM), in order to configure the new AMF instances.

4.3.1 Traffic steering

As the exchanged messages between the network functions, in the NGC architecture, are not defined yet by the 3GPP group, we assume that the legacy messages will be adapted for the new entities. Indeed, in LTE, at the end of the association procedure between the eNodeB and the MME, the MME sends its relative capacity to the eNodeB. The relative capacity of an MME is a value between 0 and 255, which allows the eNodeB to balance the load between the MMEs in the same pool. In our solution, this message will be exchanged between the AMF and the eNodeB functions. Further, we propose adjusting the relative capacity field dynamically in order to give the eNodeB accurate information about the load of each AMF. The modification of this field uses “Configuration Update” message defined in 3GPP standard [75]. The value of

the relative capacity of each AMF will be given by the control model, described below, depending on the system load and on the state of each AMF. Thus, each AMF will maintain an optimal load in order to process the requests with no additional latency.

4.3.2 Scale In/Out

As mentioned earlier, the chapter contribution is a scale in and scale out algorithm for the AMFs depending on the overall load. In order to apply those services on the AMFs, we assume that the RAN is compatible with the DNS notification mechanism [76]. In that situation, the RAN will subscribe to the DNS list, and, thus, it will be notified of the creation and the deletion of a given AMF.

Scale out

When the control system, running in the SO, detects the need of a scale out for the AMF, the SO deploys a new AMF in the architecture. In addition, it notifies the DNS of the creation of the additional AMF. In its turn, the DNS pushes the new AMF IP to the RAN. In such a case, the control system push new relative capacity for each AMF if needed. Finally, the AMFs notifies the RAN about their new AMF relative capacity using the “Configuration Update message” defined in 3GPP standard as mentioned in Sub-Section 4.3.1.

Scale In

When the control system detects a possibility of a scale in, the SO notifies the AMF in question. Once the AMF is notified, it pushes a relative capacity value equal to 0. Consequently, the RAN will not send any new procedure to this AMF instance. Following the “Configuration Update” Message, the AMF triggers an S1 release Request with the cause “load balancing”. Logically, the other AMF instances will be notified about their new relative capacity following the decision taken for the scale in, so that in their turn, they will notify the RAN. Finally, the SO destroys the AMF instance when all procedures are done.

4.4 AMF Balancing and Scaling Proposal

4.4.1 Model description

In this section, we describe our proposal featuring AMF load balancing and scaling. This model is triggered in order to split the load over the available AMFs in the NGC. When all AMFs are fully used, our proposal takes in command deploying a/multiple new one(s) to keep the access to the NGC and balance the load over all the operational AMF, in order to reduce latency. This model works, also, in the reverse way. When many AMFs are deployed and the overall system is under-loaded, it will trigger a scale in procedure, in order to avoid wasting resources. Fig. 4.2 depicts the model of our AMF load balancing and scaling proposal. To start, for the sake of simplicity, we assume one RAN on the access side and multiple AMFs in the NGC. However, this model can be easily extended to consider the multiple RANs case.

The different parameters of the model are described below:

- λ: The number of arriving UE requests
- θ: The number of UE requests that have not been satisfied and that need to be re-sent by the UEs
- w: The number of UE requests in the buffer waiting to be dispatched over the deployed AMFs
- q_i: The number of UE requests that are sent to AMF i for processing
- m_i: The number of UE requests being processed by AMF i
- μ_i: The number of UE requests satisfied by AMF i.

Following the generic system parameter description, the state evolution of our model

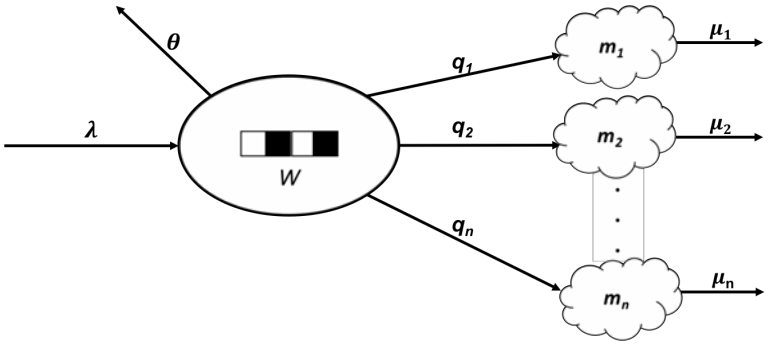


Figure 4.2: System Model

is written below following the discrete-time system of equations:

$$\begin{cases} m_1(k+1) = m_1(k) + q_1(k) - \mu_1(k), \\ m_2(k+1) = m_2(k) + q_2(k) - \mu_2(k), \\ . \\ . \\ . \\ m_n(k+1) = m_n(k) + q_n(k) - \mu_n(k), \\ w(k+1) = w(k) + \lambda(k) - \theta(k) - \sum_{i=1}^{n} q_i(k), \end{cases} \tag{4.1}$$

Let $p_i(k) = q_i(k) - \mu_i(k)$ and $\gamma(k) = \lambda(k) - \theta(k) - \sum_{i=1}^{n} \mu_i(k)$. The system represented in (4.1) can be reformulated as below:

$$\begin{cases} m_1(k+1) = m_1(k) + p_1(k), \\ m_2(k+1) = m_2(k) + p_2(k), \\ . \\ . \\ . \\ m_n(k+1) = m_n(k) + p_n(k), \\ w(k+1) = w(k) + \gamma(k) - \sum_{i=1}^{n} p_i(k), \end{cases} \tag{4.2}$$

The model (4.2) can be written as a discrete-time linear system, in the form:

$$\begin{cases} X(k+1) = AX(k) + BU(k), \\ Y(k) = CX(k), \end{cases} \tag{4.3}$$

where, the state vector

$$X(k) = [M_n(k) - M_n^{ref}, w(k) - w^{ref}]^T$$

and

$$M_n(k) = [m_1(k), m_2(k), ..., m_n(k)]$$

The constant vector $M_n^{ref} = [m_1^{ref}, m_2^{ref}, \ldots, m_n^{ref}]$ represents the targeted load of the AMFs. This will avoid AMF overhead and guarantee that requests are satisfied with a limited latency. w^{ref} is the targeted overload allowing to detect if an AMF scale in/out is needed for the system to guarantee an optimal requests' dispatching and processing while minimizing resources' wastage.

The control vector $U(k)$ is defined as follows:

$$U(k) = [p_1(k), p_2(k), ..., p_n(k), \gamma(k)]^T$$

The remaining matrices are, thus, defined as follows:

$$A = C = I_{n+1},$$
$$B_{(n+1)\times(n+1)} = \begin{pmatrix} I_n & 0_{n\times 1} \\ -1_{1\times n} & 1_{1\times 1} \end{pmatrix}$$

The output $Y(k)$ of this system represents the load of each AMF and the state of the buffer at time step k.

It can be checked easily that all the eigenvalues of the matrix A do not satisfy the stability condition[1] cf. [77]. This means that the system described in (4.3) is unstable and does not converge to the desired state, if no control action is performed.

The controllability [78] of this model can be analyzed by calculating the controllability matrix, which is defined as follows:

$$C = [B \quad AB \quad A^2B \quad ... \quad A^{n-1}B]$$

To be controllable, the controllability matrix of the system should have a full row rank. Indeed, for our model, the controllability matrix has a full row rank equal to n. It can, also, be checked that the system is observable [78].

[1] All eigenvalues should be strictly smaller than 1.

4.4.2 Dynamic AMF Load Balancing

We focus in this part on the design of the regulator's model to stabilize the whole system by scheduling the UE requests to a given AMF, with the objective to efficiently use the available resources. Also, it will control the decision of AMF scale out in case where more resources are needed, or a scale in when fewer resources are used to reduce resources' wastage. It is worth recalling that the regulator's model is a function that runs at the SO.

Since the controller, following the requirements listed above, needs to dynamically and in real-time calculate m_i, θ and w, it will be based on the Linear Quadratic Regulator (LQR) [79]. More specifically on infinite-horizon, discrete-time LQR. In LQR model, there are two main characteristics: the performance index J and the feedback vector $U(k)$. The performance index is given as follows:

$$J = \sum_{k=0}^{\infty} [X(k)^T Q X(k) + U(k)^T R U(k)], \tag{4.4}$$

where $X(k)$ and $U(k)$ are the state vector and the feedback (control) vector, respectively. The LQR aims to minimize the performance index in order to allow the system to converge to the goal with less controller action. Therefore, Q and R from 4.4 as well as the cost matrices should satisfy:

$$Q = Q^T \geq 0, R = R^T > 0 \tag{4.5}$$

Finally the feedback vector $U(k)$ can be written as the following:

$$U(k) = -KX(k) \tag{4.6}$$

where the matrix $K = [R + B^T S(k+1)B]^{-1} B^T S(k+1)A$ and the matrix $S(k)$ is the solution of the following Riccati difference equation [79]:

$$\begin{aligned} S(k) = Q + A^T S(k+1)A - \\ A^T S(k+1)B[R + B^T S(k+1)B]^{-1} B^T S(k+1)A. \end{aligned} \tag{4.7}$$

In steady state, $S(k) = S(k+1) = S$, thus, Riccati equation expressed in 4.7 can be written as:

$$S = Q + A^T SA - A^T SB(R + B^T SB)^{-1} B^T SA \tag{4.8}$$

The optimal control can thus be described by:

$$U(k) = -(R + B^T SB)^{-1} B^T SAX(k). \tag{4.9}$$

4.4.3 Dynamic AMF Scaling

Having detailed the AMF load balancing algorithm, we focus now on the AMF scaling process. Thanks to the control vector $U(k)$, the buffer $w(k)$ is calculated and updated in real-time. In addition to those two parameters, new parameters, inspired from [80], will be calculated. Starting by the Average Loss $aLoss$ that is determined as follows:

$$\textit{aLoss}(k+1) = \textit{wf} \times \textit{aLoss}(k) + (1 - \textit{wf})\frac{\theta(k)}{\lambda(k)} \tag{4.10}$$

where *wf* is a weight factor. Once we have the *aLoss*, a Congestion probability is deduced as follows:

$$\textit{CongP} = \textit{aLoss}(k) + \beta \times \sqrt{\textit{aLoss}(k)} \times \gamma(k) - w^{\textit{ref}} \tag{4.11}$$

where β is a learning factor.

If *CongP* is higher than a fixed probability threshold, then a scale out is needed. Thus, a new AMF is deployed and integrated to the whole system. Otherwise, a scale in possibility is studied. For that, we need to check if the average load of the actual system can be supported if we remove one AMF. If it is the case a scale in procedure is triggered.

4.5 Model Evaluation and Results Analysis

4.5.1 Test Scenarios

As stated before, the first objective of our model is to dynamically scale out or scale in the AMF. The second one consists on dispatching the UE requests intelligently in order to have an optimal load in each AMF and, thus, decreasing the response latency. To test and validate our model, four scenarios were implemented using Matlab, in addition to the probabilistic EWMA. The EWMA model consists on dispatching the UE requests randomly over the deployed (active) AMFs. The envisioned scenarios will

allow us to evaluate the performance of our model, which is based on control theory, against the EWMA model.

For both control model and EWMA model, we fixed the highest number of AMFs (5 AMFs) that can be deployed during the scenarios. Each AMF supports a maximum load of up to 30 processes by time step. Above this highest rate, the UE arrivals will be blocked. We fixed also the optimum AMF load rate to 20 requests by time step. For this optimum rate, the AMF can process the requests with no latency. Above this rate, an additional latency, which may take very high values, is added to each request.

The first scenario implements different arrivals' rate, based on Poisson distribution, over a given period of time. The arrival rate will be increased over the time until reaching the maximum capability of the system, and, then, it will be decreased until reaching a minimal arrival rate. This scenario allows to validate the dynamic adaptation of our model and its flexibility by scaling out the AMF NFs when the arrival rate increases and by scaling in when the arrival rate decreases. This scenario will also prove the stability of our model and will show the ability of our model to keep the AMF load around the optimal processing value.

The three other scenarios represent an under-loaded system, a fully-loaded system and an overloaded system. Each scenario is repeated 30 times in order to compute confidence intervals. Those scenarios will allow to see the behavior of our model following different load patterns and to compare it to the behavior of the EWMA model.

4.5.2 Results Analysis

Following the scenarios' description, in this section we discuss the results obtained for each of them. The results of the first scenario are depicted in Figure 4.3. Following the arrival dispatching plots, we notice that the EWMA model schedules the requests randomly over the five AMFs (q_1 to q_5) independently from the arrival load ($lambda$), unlike the control model. From the AMF load level plots, we deduce that when the arrival rate increases, the EWMA model pushes the AMFs loads (X_1 to X_5) to their limits (30 process at a time), hence adding latency for each process. However, the control model dispatches the arrivals in a manner to have an optimal load (20 processes at a time) in each active AMF and thereby processing the requests with no additional latency. From the Number of Active AMF plot, we notice that the EWMA is not stable and falling in what is called a Zeno phenomenon ("ping-pong"). The control model shows more

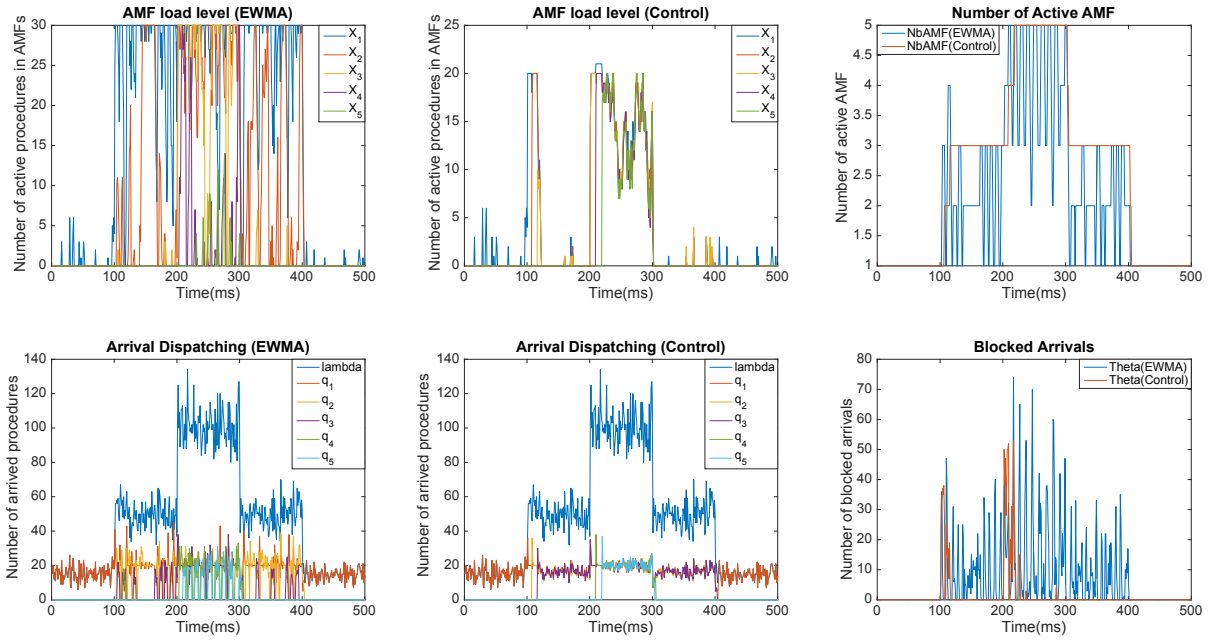


Figure 4.3: Evaluation of the Adaptation Of the Control Model

stability and scales in/out the AMF instances accurately depending upon the needs.

The results of scenario 2 to 4 are depicted in Figure 4.4. Thanks to those scenarios we are able to compare the behavior of the control model, and the EWMA model in three traffic patterns: under-loaded system; fully-loaded system and overloaded system.

In the under-loaded system plot, we notice that the EWMA model is using three AMFs as the control model. However, the EWMA load (X) is not balanced between the three AMFs, while the control model has an equivalent load over the three AMFs.

In the fully-loaded scenario, we notice that the EWMA model scheduled randomly the arrivals over the five AMFs with an average of 28 processes at a time. Thus the EWMA model is loading the AMFs more than the optimal AMF load, and thereby adding latency while satisfying the arrivals. In contrast with the latter scenario, the control model transfers the arrivals to the five AMFs depending on their load in order to maintain the optimum AMF load when possible and so satisfying the requests with no additional latency. Despite the following scheduling solution, it is important to notice that the average number of blocked arrivals, in this category of tests, for the control model is lower than the EWMA model. Therefore, the control model is able to distribute the load among the AMFs without additional latency nor blocked arrivals.

Finally, in case of overloaded conditions, unlike the EWMA model, the control model avoids fully loading the AMF, and tries to keep its load around the optimal AMF load. Thus, some additional arrivals will be blocked as shown on the average of the blocked arrivals' plot.

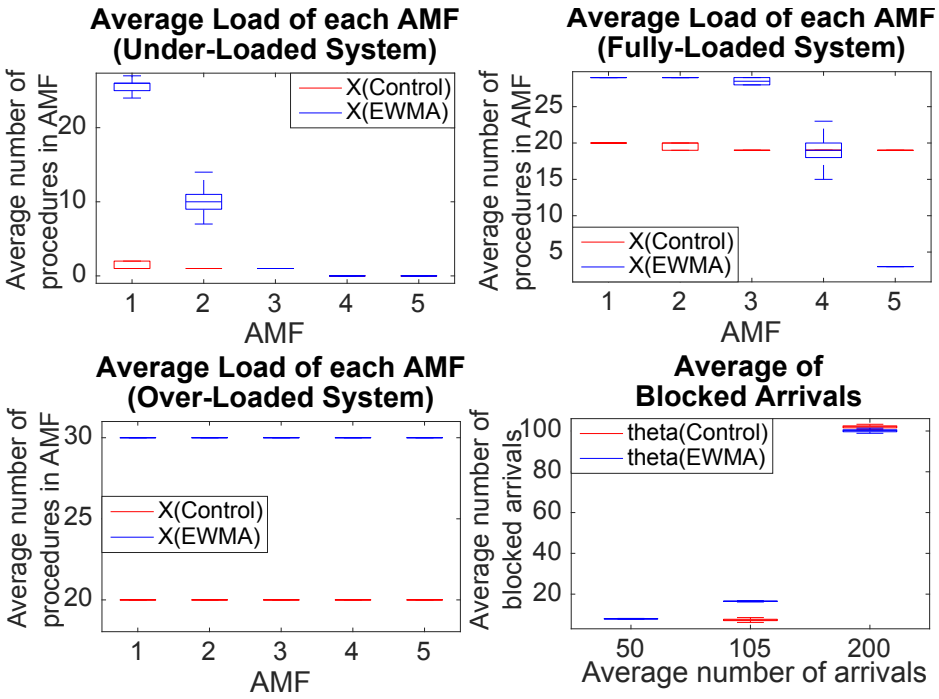


Figure 4.4: Evaluation of the control model Behavior vs different arrival Load

To summarize, the control model can schedule the arrivals depending on the AMFs load in order to maintain the optimal AMF load and to satisfy the request arrivals with no additional latency, especially as decreasing latency is an important requirement for 5G. Furthermore, the control model is designed to avoid resources' wastage by only activating (deploying) the exact number of needed AMFs. Indeed, it can scale out the system in order to satisfy an arrival rate increase, if needed, and scale in when arrival rate decreases.

Unfortunately, reducing latency and scaling dynamically add some penalties on the system. As mentioned above, some sessions will be blocked when the system is over-loaded or when scaling up. However, the system can remedy from this penalty by adding a prediction function allowing to deploy in advance a new AMF when arrival rate is predicted to increase.

4.6 Conclusion

In this chapter, we introduced an algorithm based on control theory in the scope of NGC work of 3GPP group. This algorithm allows to steer UE control traffic according to the AMF load, which maintains an optimal load (i.e. no UE blocked) in AMFs' instances. Additionally, we showed that the design of this control helps saving resources by scaling out and scaling in the AMF capacity as needed. Besides, procedures are proposed to be able to deploy the algorithm in the NGC architecture following the 3GPP standards.

Although congestion control is good, congestion avoidance would be better. We anticipate that Machine learning techniques may be used in order to predict the upcoming load of the core network and thus to avoid congestion by scaling in advance. Therefore in the next chapter we will introduce a congestion avoidance mechanism in the scope of 5G core network, based on Machine Learning and more precisely on neural networks.

CHAPTER 5

CONGESTION AVOIDANCE IN NEW GENERATION CORE NETWORK OF 5G

5.1 Introduction

5G is expected to provide network connectivity to not only classical devices (i.e. tablets, smartphones, etc) but also to the IoT, which will drastically increase the traffic load carried over the network. 5G will mainly rely on NFV and SDN to build flexible and on-demand instances of functional networking entities, via VNFs. Indeed, 3GPP is devising a new architecture for the core network, which replaces point-to-point interfaces used in 3G and 4G, by a producer/consumer-based communication among 5G core network functions, facilitating deployment over a virtual infrastructure. One big advantage of using VNFs, is the possibility of dynamically scaling, depending on traffic load (i.e. instantiate new resources to VNFs when the traffic load increases, and reduce the number of resources when the traffic load decreases). In this chapter, we propose a novel mechanism to scale 5G core network resources by anticipating traffic load changes through forecasting via Machine Learning (ML) techniques. The traffic load forecast is achieved by using and training a Neural Network on a real dataset of traffic arrival in a mobile network. The proposal is tested via discrete event simulations using a traffic load dataset provided by a Network Operator. The results show clearly the robustness of our proposal compared to a threshold-based scaling technique. Furthermore we propose a novel solution to generalize neural networks while accelerating the learning process by using K-means clustering, and a Monte-Carlo method. We benchmarked multiple types of deep neural networks using real operator's data in order to compare their efficiency in forecasting the upcoming network load for dynamic and proactive resources' provisioning. The proposed solution allows obtaining very good predictions of the traffic evolution while reducing by 50% the time needed for the learning phase.

5.2 Related Work

The introduction of the NGC by the 3GPP group provides the flexibility to scale the infrastructure much more easily to respond to increasing traffic loads or even to free up resources in the event of a drop in needs. Indeed, as the components of the NGC are run as VNFs (using Virtual Machine - VMs or containers), the scalability issue is now more a VM (container) deployment issue than a hardware deployment issue. In addition, the AMF is the entry point of the 5G core network; which makes it the system bottleneck. Thus, a special focus should be given to the dimensioning of the VNF(s) hosting the AMF to ensure system scalability. Therefore, new strategies and techniques should be envisioned for the dynamic scaling of the core network in order to adapt dynamically the resources provisioning to the needs. Thus, it is crucial to decide accurately when to scale and how many instances to deploy or to shut off. Many solutions are proposed in the literature for VNF scaling.

One of the most common scaling solutions consists in using static thresholds. In [81], [82] and [83] authors have proposed scalability based on thresholds. In the same spirit, we worked also on threshold based scalability in a previous attempt to manage AMF instances in [13]. The main idea is to deploy a new instance (scale-out) when the CPU usage (or other resources) exceeds a certain fixed “scale-out” threshold. Reciprocally, an instance is shut off (scale-in), when a considered threshold exceeds a fixed value. Static thresholds-based solutions are already deployed and widely used in commercial and open-source cloud solutions, like [84] and [85]. However, static thresholds-based approaches are not suitable for mobile networks, and have several limitations due to the fact that they react to events (i.e. reactive solution). Indeed, deploying a new instance of a VNF, when reacting to events, may take from one minute to a dozen of minutes depending on the data center architecture and the complexity of the VNF instance to scale; leading to delay the scaling-out process, hence increasing the attach duration of UE and rejection of some requests.

An alternative to static-threshold-based solutions is the usage of adaptive techniques, as proposed in [86]. In this work, the authors have proposed a mechanism that combines Q-Learning with Gaussian Processes-based system models, which allows to adapt to dynamic environments and improve the scaling policy before taking any action. The authors assume that their proposal reacts better than static threshold rules. However, it remains a reactive solution, and hence inherits the same weaknesses.

Moreover, using Reinforcement learning, it may take a considerable time before starting to have the correct decisions, which is not acceptable in systems like 5G, due to the time constraint of certain type of applications. Note also the side effects associated to this type of approaches, consisting in the influence of the external environment on performance and therefore the risk of quality degradation.

Proactive solutions for scalability have been also proposed in the literature. The authors, in [87], suggested to pro-actively schedule resources for VNFs based on CPU usage forecasting from a historical dataset. Although this technique is interesting for its pro-activity, it does not offer adaptation in case of traffic pattern evolution and is not compliant with mobile networks requirements. Indeed, the proposed solution considers only system level information, like consumed CPU or memory, ignoring service level information. VNF scaling in mobile networks should take into account strict rules as defined in standards and therefore deploying only a new instance by a cloud orchestrator is not sufficient for scaling an end-to-end service. Consequently, mobile networks require service level information, like the number of connected users or traffic load when scaling-out or -in. Moreover, the parameters of the "offline" prediction equation, proposed by the authors, are determined statically and are completely independent from the considered dataset. This implies stability in the weights that are given for each day prediction, and thus a risk of divergence in the prediction with the slightest change of pattern. Concerning the "online" approach, the authors propose to use an average (exponential average) estimator. However following [88], when a pattern can be determined, this is clearly not the best solution.

Finally the authors in [89], worked on accelerating neural networks training using clustering. The main objective of the paper was to use k-means clustering to reduce data size for the training phase while keeping almost the same accuracy with a shallow neural network. However, the analysis of the results did not go far enough to show the amount of samples needed to converge the classification problem under consideration. Indeed, no clear results were presented on how much the technique can accelerate the training process or even its efficiency. Moreover, in high dimensionality, k-means is not that efficient and the determination of the number of samples remains an open issue. On the other hand, reducing the number of samples results in a loss of information that could severely affect the quality of learning. This is one of the key points that we are improving in this chapter by resorting to a Monte-Carlo method.

5.3 Architecture

As mentioned in 4.3, the 3GPP group introduced the NGC in order to answer the requirements of 5G. In what follows in this chapter, we consider this same architecture depicted in Figure 4.1.

5.4 Basics on Neural Networks

Before stating the proposals, a brief review of some types of Neural Networks that have been used to elaborate the contributions of this chapter are presented in this section.

5.4.1 DataSet

The main key of prediction using deep learning, is the dataset. Nowadays, operators are able to collect those data easily as they are the owner of their infrastructure and have a clear idea of the evolution of the load over their network. In this work, we have used the dataset collected during the Big Data Challenge organized by Telecom Italia[90]. The data set is open source [91] and rich in information. It can be used to predict the evolution of the load in a core network.

The considered data set contains several kinds of data, like sms in/out, call in/out and the Call Detail Records (CDRs) with an interval of ten minutes for two months; including information on both control plane and data plane traffic. We consider in this work that at least 10% of the traffic is control plane traffic, and the remaining 90% is data plane traffic. Therefore, for the remainder of this chapter, we will only use the control plane traffic (10% of the load in each period of ten minutes) for prediction.

5.4.2 Neural Network Types

Deep Neural Network

The DNN, or also called feed forward neural network, is composed of several neural layers. The DNN layers are classified as following: the input layer; the hidden layer(s); and the output layer. Each Layer is composed of nodes. Each node is like a neuron in our brain: Following a stimulus it may fire an action. The stimulus is actually the data X

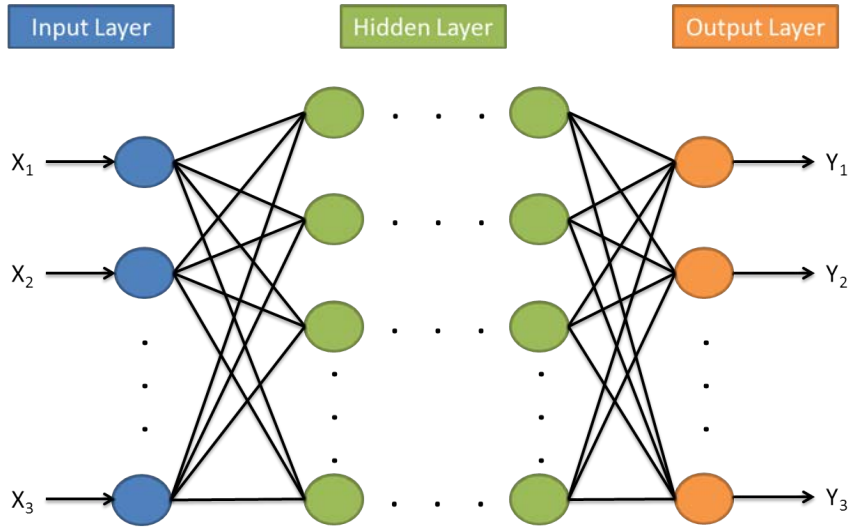


Figure 5.1: DNN architecture

multiplied by a weight. Once we have stimuli, the stimuli are summed and passed to a function called 'The activation function'. The activation function detects the importance of the data and decides whether the data is propagated through the network in order to get the final outcome or vanishing it.

Recurrent Neural Network

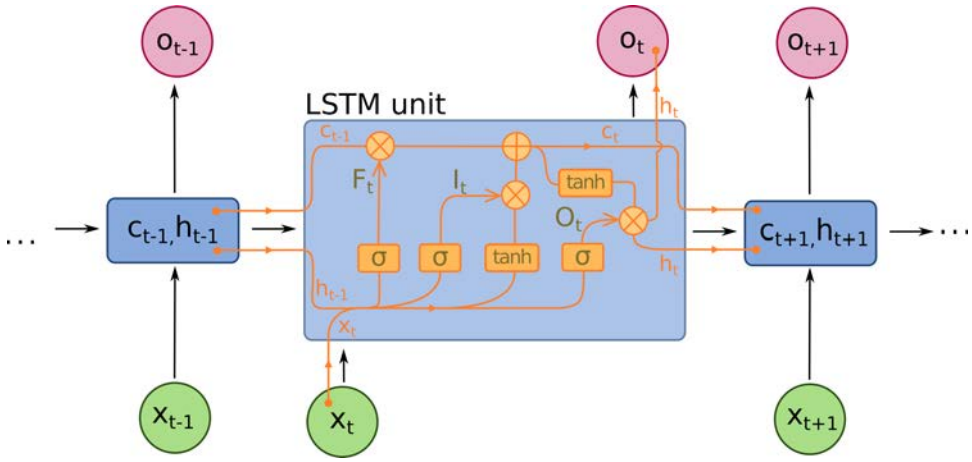


Figure 5.2: Long Short Term Memory Cell Architecture

The RNN is another type of neural networks. In our contribution, only LSTM cell [92] is used. The main difference between feed forward neural networks (i.e. DNN)

and LSTM, is that the RNN keeps state memory of the last passed activation events in the network as temporal contextual information [93]. Figure 5.2 illustrates the structure of an LSTM cell. An LSTM cell is composed mainly of an input and three gates: the input gate, the forget gate and the output gate.

The input x_t is squeezed between -1 and 1 using the $tanh$ function and can be noted as following:

$$g_i^{(t)} = tanh(b_i^g + \sum_j U_{ij}^g x_j^{(t)} + \sum_j W_{ij}^g h_j^{(t-1)})$$

i and j respectively represent the cell and the neuron. b, U and W respectively denote the biases, input weights and recurrent weights into the LSTM cell. Then, the output of the input gate (a sigmoid gate), which is between 0 and 1 (I_t), is multiplied element-wise by $g_i^{(t)}$ in order to determine which inputs are switched on and off. This operation is denoted as $g \circ i$. Afterwards an inner state (C_{t-1}), providing an internal recurrence loop to learn the relationship between inputs separated by time, is ultimately added to the $g \circ i$.

Furthermore, the forget gate, a sigmoid function, is element-wise multiplied by C_{t-1} to determine which previous states should be remembered and which should be forgotten, allowing the LSTM cell to learn appropriate context. Finally, the output O_t of the sigmoid gating function is multiplied by the squeezed inner state s_t in order to get the LSTM cell output expressed as following:

$$h_t = tanh(s_t) \circ O_t$$

For more details on RNN and LSTM structures and functions, interested readers may refer to [94].

Convolutional Neural Networks (CNN)

A CNN, depicted in Figure 5.3 (CNN), comprises three main layers. The convolution layer is used to extract features from the input data, while preserving its spatial relationship. Then the non-linear operation Rectified Linear Unit (ReLU)[1] is applied. Following the ReLU function, spatial pooling (sub-sampling) is applied in order to reduce the dimensionality of each feature map while keeping the most important information. Finally

[1] ReLU: $f(x) = max(0, n)$

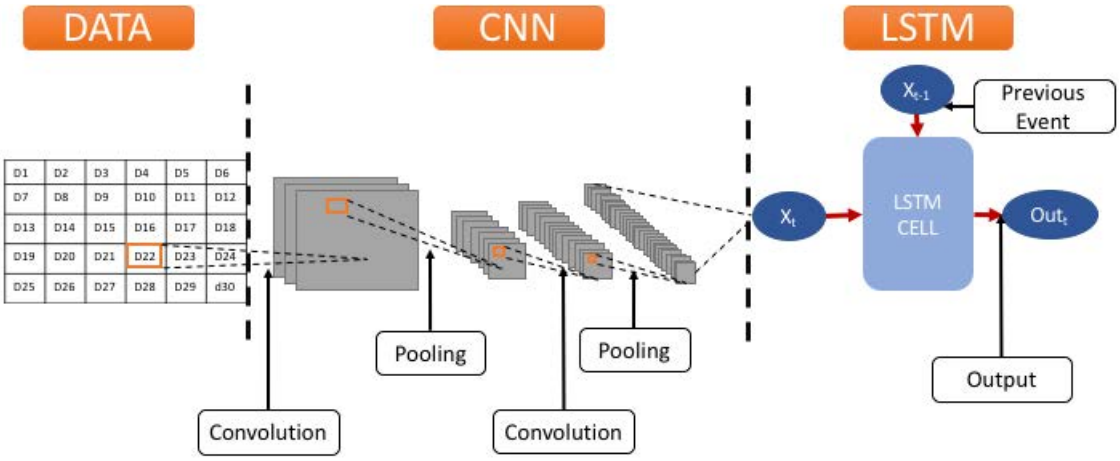


Figure 5.3: CNN-LSTM Architecture

and before the output, a final fully connected layer is used. This final step will allow to classify the output and to learn non-linear combinations of the features (refer to [95] for more details).

A combined CNN–LSTM architecture

In addition to the two types of neural networks explained above, we added another one by combining both types used before. This third type will allow getting improved results by benefiting from the strengths of both the LSTM and the CNN. Indeed, as a an entry we use the CNN in order to extract features (i.e. traffic pattern) and simplify them by using the pooling, which will allow retaining the main part of the pattern. Hence, beside using a fully connected layer as an output layer, we use the LSTM cell in order to forecast the upcoming traffic load.

5.5 ML-based scalable mechanism for 5G core network

5.5.1 Traffic forecasting: Deep learning-based approach

As explained above, there is a need of a mechanism to allow dynamic dimensioning of the AMF resources to avoid degrading the system QoS. The envisioned mechanism should be proactive to avoid the weakness of reactive solutions; hence requiring a traffic prediction solution. For this aim (i.e. traffic prediction), two Neural-networks-

based techniques were envisioned and tested aiming at predicting the upcoming traffic load. The first one is based on Deep Neural Network (DNN), while the second one relies on RNN, more specifically on LSTM cell. In a DNN, a particular neuron can be activated by the contribution of several inputs at once, and therefore there is no real recognition of a traffic pattern. On the contrary, a recurrent neural network, and particularly an LSTM network, receives the inputs successively and through a memory and omission mechanism, it manages to memorize traffic patterns, as it has been demonstrated in several works [96].

The main procedures for traffic prediction using learning techniques are: getting the dataset, formatting the data, designing the neural network, training the neural network and then starting predicting (or testing your network). Each step is important in order to get the best accuracy of prediction compared to real values.

Mobile Dataset and Data Classification

Once the dataset is collected and 10% of each period is extracted, the data should be formated in order to be injected in the neural network for the training phase. This part requires a high attention as the data format can affect the accuracy of prediction of the neural network. Formatting intelligently the data will allow the neural network to learn more in the training phase, and thus predicting with a high accuracy. However, adding lots of KPI may lead to decreasing the prediction accuracy further, particularly for small datasets. Therefore a balance should be found between the dataset size and the KPI injected in the data formatting.

Once we have the needed data, we propose to shift from a prediction problem into a classification one. This means that at each time stamp, the load data will belong to a load class. The load classes are distributed as following: the variation between the minimum control load and the maximum control load over the two months is split into 10 intervals. Thus the period between each interval of load is a class. Therefore, depending on the class, we are able to determine how much AMF instances we need, as we know already the capacity of our AMF (20 process per second in our case).

Then, each class is coded in order to increase the learning factor and the accuracy of our neural networks. The coding of the classes is simple and based on 'one-hot' coding. In other words, the class are coded using a word of 10 bits (as there are 10 classes), and the first class code will be the first bit equal '1' and the others equal '0'. The second class will have the second bit equals '1' and the other equal '0'... etc.

Finally to sum up, at the end of the data processing and classification, at each time stamp over two months, instead of having a number of connections, a class of load is defined where the number of connections fits in the interval of the class.

Note that adding a much larger number of output classes would require a much larger dataset. Indeed, the addition of classes implies the increase of the number of parameters of the neural network which has an effect of requiring more data to have a significant precision.

Neural networks-based prediction

The first technique used for predicting the class of load of the next period is DNN. DNN is a neural network composed of three main layers: the Input layer, the middle layer(s) (hidden layer(s)) and finally the output layer. This type of neural network is also known as the Feed Forward Neural Network (FFNN). The data goes in the neural network from the input layer through the middle layer(s), and finally go out by the output layer.

The second technique tested for predicting the average load of the upcoming period of time is RNN, which is another type of neural network. Unlike DNN, RNN have loops. Those loops allow to inject previous events while predicting future events. It is a sort of memory block allowing to take decision depending on the previous event(s). Also, in RNN multiple kinds of cells are used today. In this chapter we chose to rely on the LSTM cell. The advantage of the LSTM cell is to allow data patterns to be stored without degradation over time.

In order to compare the accuracy of the DNN and the RNN (LSTM), one DNN and one RNN (LSTM) were designed using the Tensor Flow library[97]. We divided and formatted the data into classes of average load. Then, we split the original dataset into two separate datasets. The first one contains 60% of the data, for training the two neural networks; while the remaining 40% are left to test the robustness of the two networks. Then both networks are trained with the 60% dataset and then asked to predict the remaining 40%. Figure 5.4 illustrates the prediction of both the DNN network and the RNN network. From Figure 5.4 we notice that RNN (LSTM) network performs better than DNN network. The RNN (LSTM) network achieves an accuracy of $\approx 90\%$, while the DDN network obtains $\approx 80\%$ of accuracy; hence the RNN is performing more than 10% better than DNN when predicting the class of average load of the upcoming period of time. The accuracy of the prediction is mainly due to the ability of LSTM

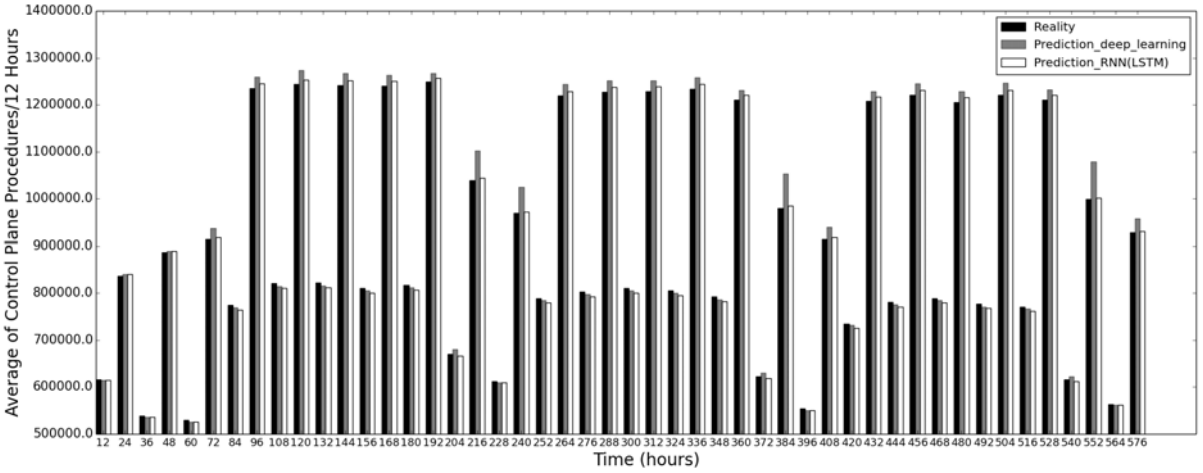


Figure 5.4: Reality vs NN prediction

networks to recognize a traffic pattern, unlike DNN networks. The accuracy of the DNN network can, however, be improved but this would require more neurons and more data, something that is not always available.

5.5.2 AMF Balancing and Scaling Proposal

Prediction-based scalability

Having described the prediction model, now we present the prediction-based scalability algorithm. Based on the ETSI NFV reference architecture, the scalability of a VNF is a decision taken by the NFVO, using the information provided by both the VIM and the VNFM. The former provides information on the VM execution environment (like CPU usage, Memory usage), while the latter provides information obtained from the EM hosted in the VNF. The information provided by EM, represents the inside VNF service performances, like the number of attach requests or the size of the queue of the AMF, etc. In [13], we proposed that the EM of the AMF reports this information to the VNFM, hence the NFVO. The latter takes decisions to scale-in or -out using predefined thresholds. However, due to the time needed to scale out (run a new VM), the proposed solution suffers from delay to react to the increase of UE attach; which leads to an increase in the time of the users' attach with some losses of requests.

To remedy to this problem, in this work, we will use the RNN to predict the needed number of AMFs that efficiently manages the traffic load, aiming at keeping the at-

tach duration under an acceptable value (using the results shown in section III.A). The NFVO uses the RNN to predict the traffic (i.e. class of traffic), hence derives the optimal number of AMF to pro-actively deploy before the change of traffic load; anticipating any network load changes and removing the time needed to start a new VM instance, in case of scale-out process. Besides, in this work there is no need to monitor the VNF performance via the VNFM, as the NFVO relies only on the trained RNN, which permits to reduce the communication load introduced by the exchanged messages between the EM and VNFM.

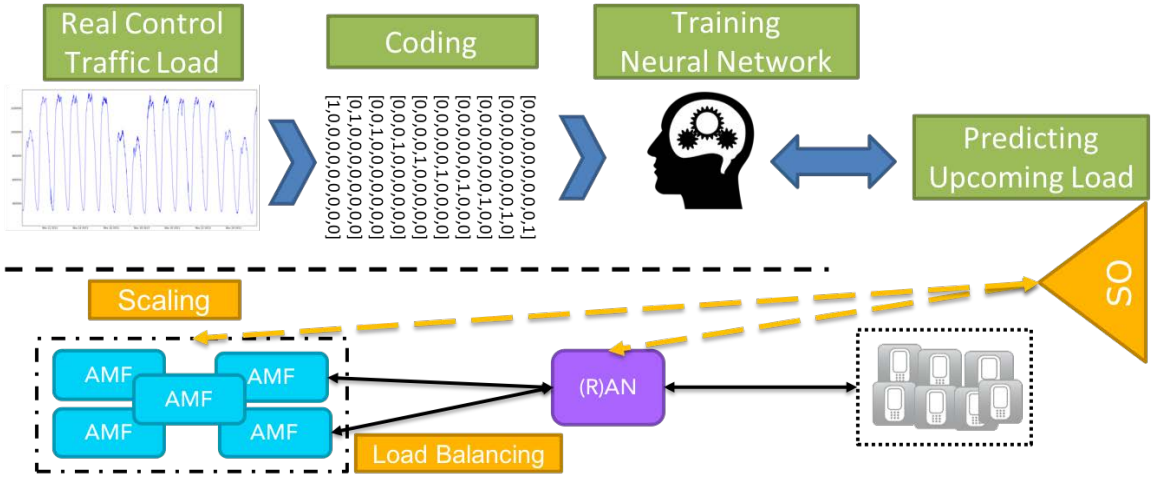


Figure 5.5: Proposal Logic

Load Balancing

To equilibrate the load between the AMF instances, the control model developed and tested in Chapter 4 will be used. The used-controller is LQR [79]. The LQR aims to minimize the performance index in order to allow the system to converge to the goal with less controller action. So periodically, the LQR will calculate a feedback vector allowing to determine the relative capacity of each AMF, depending on its load. Once the relative capacities are calculated, they are pushed to the eNodeB in order to distribute the arrivals on the AMF instances depending on the load of each.

Simulator

In order to test the models, we developed a discrete event simulator in Python using Simpy [98]. The simulator simulates the behavior of UE arrivals that are buffered in the eNodeB queue. Then, the simulated eNodeB will dispatch the arrivals on the active AMF instances based on their relative capacity calculated periodically as mentioned above. The dispatched UEs will be hosted in the corresponding AMF queue and processed following a speed of 20 requests per second. This means that the latency chosen for a given AMF instance is $0,05$ milliseconds. Also, a given UE procedure will be rejected, due to the overload of the AMF instance, if the number of requests bypasses 50 requests in the AMF queue. In addition to the relative capacity calculation for each AMF, the SO checks each 10 minutes the neural network model in order to get the load class of the next period. Depending on the predicted class, the SO determines the needed number of AMF instances to be deployed for the next period. If the number of instances will increase, the SO will deploy the instances 60 seconds before the next period. The 60 seconds are considered as deployment latency. It is the needed-time to deploy an AMF instance in the data center and to be fully operational. The deployment latency (60 seconds) is calculated as an average following the tests conducted in [99]. Otherwise, for scaling in the AMF instances, the SO will order the removal of the over provisioned AMF instances 60 seconds after the next period in order to be sure that the load have decreased and avoid AMF overload. Finally, as the collected arrival data has a periodicity of 10 minutes, we distributed each period on 600 seconds in order to get more granularity and simulate a real traffic pattern. Therefore, the total number of arrivals of each period of 10 minutes is distributed following an inter arrival time, for each UE, calculated based on Poisson process law.

Scenarios

In order to test and compare our proposals, three different simulations were conducted. For the three simulations, we have considered the same arrival data (one week) with an AMF instance capacity of 20 requests per second. Also a deployment latency is considered over all the simulations equal to 60 seconds. The relative capacity refreshment of each AMF is achieved by the SO each 10 milliseconds.

- Static Threshold Simulation: The first simulation consists on validating our simulator and showing the behavior of a dynamic scalability based on a static thresh-

old, while using our control model to balance the load over the deployed AMF instances. In addition to the simulation parameters listed above, we consider a scale out threshold of 80% of load of the overall deployed AMF capacity. The scale in procedure is launched once less than 10% of the overall AMF instances capacity is used. The scalability decision is checked each 2 seconds.

- Deep Neural Network Simulation: In this simulation we deployed a DNN composed of an input layer; 3 hidden layers and an output layer. The hidden layer 1 is composed of 500 neurons, the hidden layer 2 is composed of 1000 neurons and hidden layer 3 is composed of 1500 neurons. In order to train our DNN, we extract 60% of the dataset, shuffle them, then inject them into the DNN as batches of 500. From the 40% of data left, we selected one week of data and injected it in the simulator (the same data as the other simulations). So in this simulation, the SO checks each period of time (600 seconds) the predicted load class given by the DNN for the upcoming period of 600 seconds and decides whether to scale in/out the AMF instances.

- Recurrent Neural Network Simulation: Finally, for this simulation we deployed the RNN composed of one LSTM cell. This cell is composed 256 neurons. As the DNN, 60% of the dataset were shuffled and injected as batches of 500 in the RNN. In order to keep the value of our data classification for the RNN and get the best prediction accuracy, each training data of a batch is composed of one RNN chunk of which has the size of the word of a coded class (10). Like in the DNN case, we selected one week of data from the 40% left in the dataset and inject it in the simulator(the same data as the other simulations).

5.5.3 Models Evaluation and Results Analysis

In Figure 5.6, we compare the number of deployed AMF instances in terms of the evolution of the load for (a) the threshold solution, (b) the DNN solution and (c) the RNN solution, and also the consequences on the AMF instances load in (d) threshold-based solution, (e) the DNN solution and (e) the RNN solution.

The threshold-based solution deploys up to 13 instances when the arrival rate is at the maximum rate as the RNN and DNN solutions deploy up to 12. On the other hand, when the load is at the minimum, the threshold-based solution deploys down to

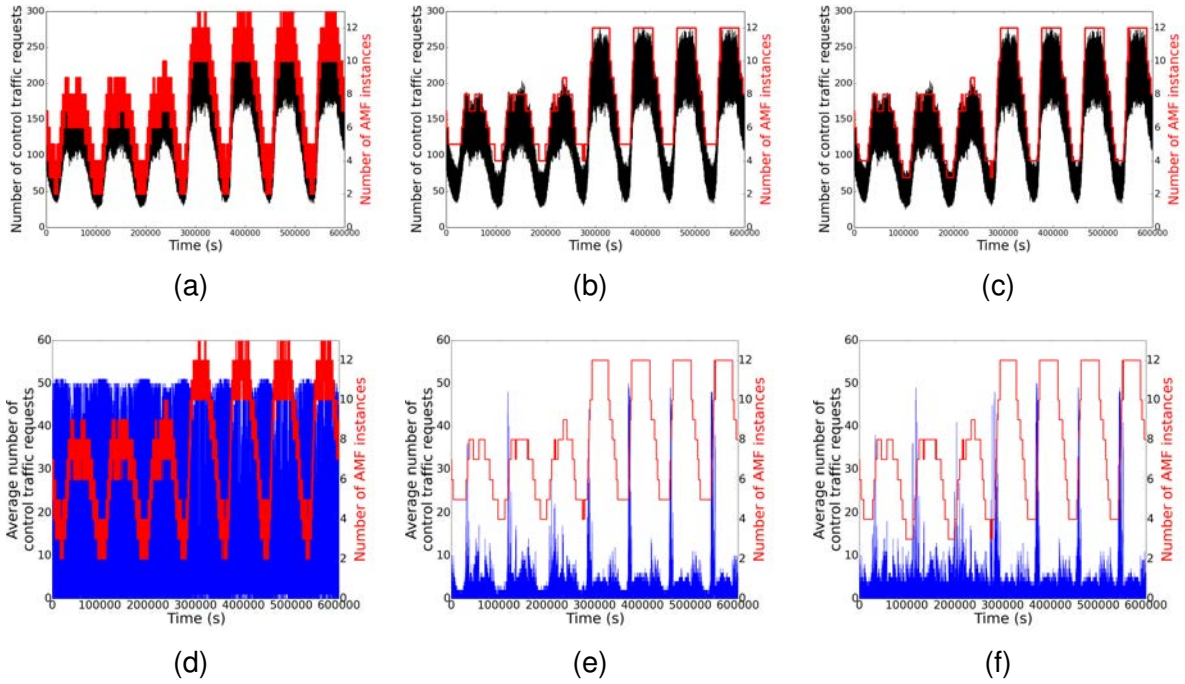


Figure 5.6: Evolution of AMF instances number with control traffic Requests (a) threshold-based solution (b) DNN-based solution (c) RNN-based solution compared to the evolution of AMF instances load with AMF instances number (d) threshold-based solution (e) DNN-based solution (f) RNN-based solution

2 instances, the DNN solution deploys down to 4 instances and finally the RNN deploys down to 3 instances. Although the threshold solution deploys more AMF instances than the RNN and the DNN solution, when the load is at its maximum, the AMF instances load is always reaching its maximum of 50 requests per seconds and thus the AMF instances are always overloaded. As for the DNN and RNN based solutions the AMF instances load is around 10 requests in the queue per seconds. This is explained by the fact that the threshold solution is reactive. Hence, it deploys the AMF instances after a load increase without foreseeing the time needed to provision new resources, which leads to AMF congestion and scaling oscillations.

On the other hand from Figure 5.7, we can see the consequences of overloading the AMF instances by evaluating the number of rejected requests: (a) Threshold-based solution, (b) DNN solution and (c) RNN solution, and the Cumulative Distribution Function (CDF) of the needed duration to process an attach request: (d) threshold solution, (e) DNN solution, (f) RNN solution. Comparing to the RNN and the DNN solutions, the number of blocked requests of the threshold-based solution is high. Also from the

CDF of the UEs attach duration, we notice that in the thresholds solution, most of ues duration are around 2.5 seconds. However, for the DNN and the RNN solutions most of ues duration are less then 0.6 seconds. Hence, scaling in a proactive manner allows to avoid an increase of the latency while processing the control plane procedures.

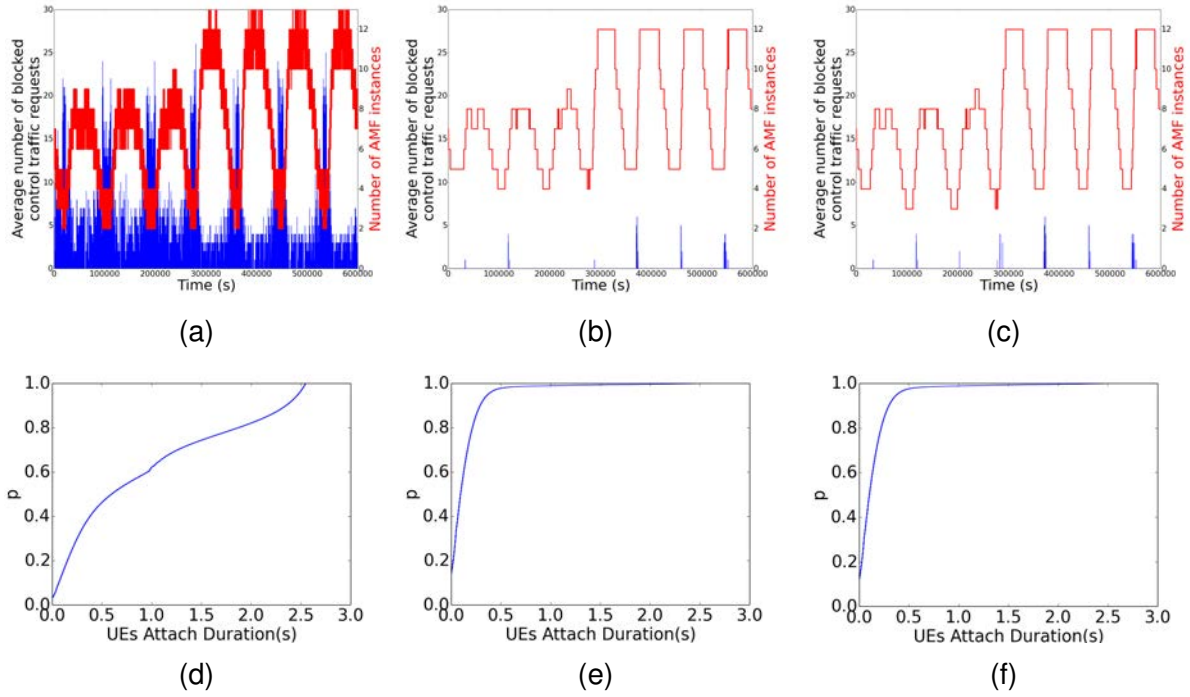


Figure 5.7: AMF instances number vs Blocked traffic Requests (a) threshold-based solution (b) DNN-based solution (c) RNN-based solution compared to the evolution of the UEs attach duration (d) threshold-based solution (e) DNN-based solution (f) RNN-based solution

Although the DNN is approximately similar to the RNN, it is worth mentioning that the RNN solution is slightly better that the DNN solution as it predicts more accurately (10% better than the DNN solution). This is shown in Figure 5.6, where we remark that when the load is at minimum, the RNN based solution scales in down to 3 whereas the DNN solution scales in down to 4 AMF instances. Indeed, the RNN predicts more accurately and thus, helps in reducing the number of AMF instances to exactly what is needed in order to absorb the actual load.

Summary

To sum up, the RNN(LSTM) technique performs better than both the DNN technique and the threshold technique. Firstly it acts in a proactive way unlike the threshold technique. Secondly it allows to deploy accurately the exact needed number of AMF instances to absorb the upcoming load. Finally, it allows to remove the deployment latency by scaling out in advance the needed AMF instances. In that way, the mobile core is ready to absorb the upcoming load increase without suffering from overhead and extra latency for the control procedures.

5.6 Neural networks traffic load estimation for resources' scaling in 5G

The prediction of the evolution of a time series, whether periodic or not, has been an active research topic for several years [100]. Several studies, in this sense, have shown the potential of neural networks, and especially LSTM, to identify traffic patterns [101].

In this section, we analyze different types of neural networks for predicting load evolution in a network. We use traditional neural networks, CNN and LSTM, but we also suggest a combined architecture of them, which not only reduces learning time but also makes it more accurate in prediction. In addition, we introduce a novel technique, based on the Monte-Carlo method, allowing to generalize the neural network to allow them to effectively predict traffic evolution even when the scale of traffic changes. The proposed technique also has the advantage of making learning much faster thanks to the use of K-means clustering technique on the data.

5.6.1 Generalizing neural nets and accelerating the training

The generalization of neural networks during learning is an important characteristic, since it makes it possible to avoid the phenomenon of over-fitting, which leads to poor performance with new data.

Various techniques have been introduced in the literature to avoid the over-fitting problem. In this chapter, we propose a novel approach that has the advantage of accelerating significantly the speed of learning. This technique is particularly effective

when predicting time series due to redundancy in the data. The main idea consists first in grouping the data using the K-means clustering technique.

In the proposed approach, we deal with windows of traffic loads, making the Euclidean distance, which is generally used as a distance metric for the clustering, not very efficient. To that purpose, we propose in this chapter using the Dynamic Time Warping (DTW) algorithm [102], which is more suitable for time series comparison.

Existing approaches proposed to reduce the size of the initial data after the clustering phase. This speeds up learning but reduces the accuracy of prediction, as only a sub-set of the initial data is used for the training. In contrast with these approaches, we propose to exploit all this data using a Monte-Carlo method.

The proposed solution consists in getting randomly, from each cluster, an element for the training phase, using the function "getRandom()". This process is repeated at each epoch, which allows covering the whole dataset and thus not loosing any precision and accuracy while accelerating significantly the training. This algorithm process has a polynomial complexity.

Algorithm 1 Monte Carlo-based training procedure

```
1: Inputs: epochs, dataset, nClusters
2: Clusters[] = K-means( nClusters,dataset)
3: for ep in epochs do
4:     trainset = []
5:     for clst in Clusters do
6:         trainset.add(getRandom(Clusters[clst]))
7:     Train the neural network with trainset
```

Note that the obtained performance is directly related to the number of clusters *nClusters*. Indeed, this number must be related to the percentage of data to be considered in each period. The more similarities we have in the input data, the smallest is the number *nClusters*.

We also propose a boost process, in which all the dataset is injected, can also be considered for a number of epochs at the end of the training phase. Based on our experiments, this phase is efficient only when the number of clusters is too small.

5.6.2 Models Evaluation and Results Analysis

Dataset

The dataset used in our work contains real mobile data collected by Telecom Italia Big Data Challenge [90] and hosted on Harvard Dataverse [91]. This dataset is rich of information. It contains two months of data collected on different Cells of the mobile core network of Mobile Italia in Milano, with a period of 10 minutes. First, the data is sorted following the timestamps and not the cell ids. In other words, for each period of time (10 minutes), we regrouped together the data of all the cells in Milano city. Then, we split the obtained dataset into two datasets: training dataset; and testing dataset. The training dataset will be constituted of 60% of the data and the testing dataset will be filled with the 40% left.

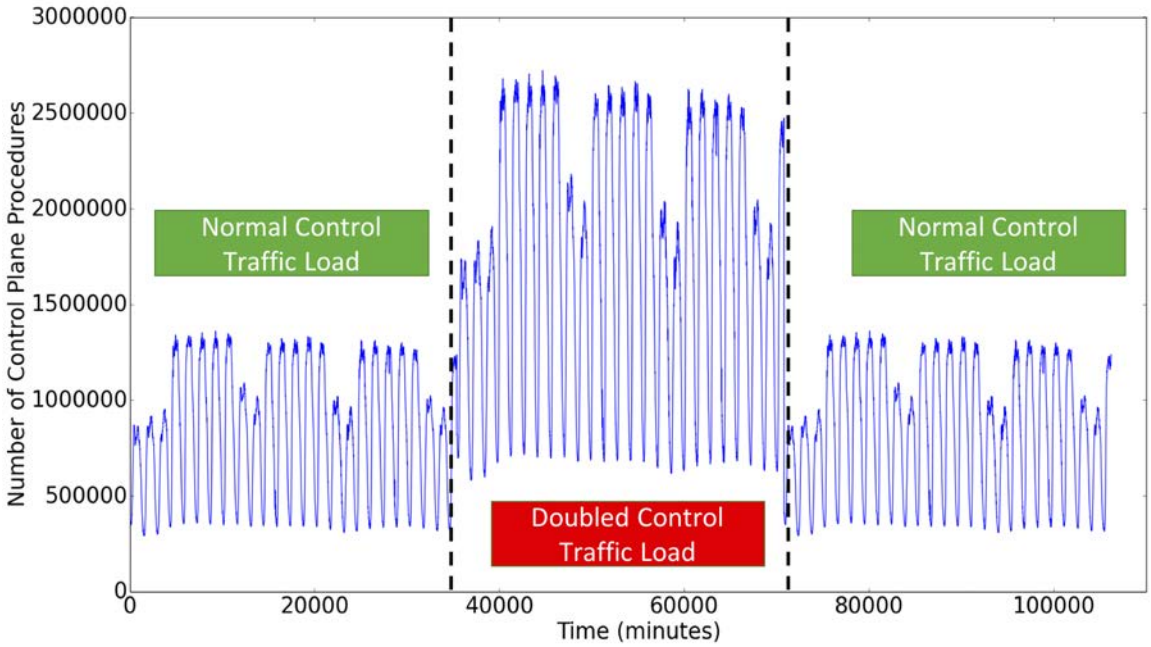


Figure 5.8: Testing data used to test the Neural Networks

Finally, we decided to add for the testing dataset, a copy of the 40% but doubled, and then also another copy of the 40%. Therefore the testing dataset, illustrated in figure 5.8, is split into 3 phases: Normal control traffic; an increased (doubled) control traffic; and return to a normal control traffic. By this, in the simulation section we will be testing the precision of the prediction of the neural networks on arrivals where the neural network is not trained for and hence testing its adaptation to traffic load pattern

change.

Scenario and simulations

In this section we explain the procedure and the simulation launched in order to validate the performance of the proposed training strategy. We built the three types of neural networks listed above using the Tensorflow library [97].

In our simulations we decided to test the effects of the number of epochs, the number of clusters (i.e. percentage of the data to consider), and the effects of the boost process. The boost technique consists on adding additional training epochs at the end of the training phase. In those additional epochs, the whole training dataset and not only the clustered data are injected. This test will allow to show if the K-means clustering based strategy is enough to get the same precision as if we inject the whole data or still a complete re-injection of the training data is needed.

For each neural network three different types of simulations are launched. In each type of simulation only the concerned parameter of the simulation type (from the parameters listed below) will vary. This helps understanding the parameter effect on the training phase, and hence the precision of the prediction and the duration of the training phase. Also, in each simulation, each entry of the neural network is filled with the last 20 past data in order to predict the 21st one. It is worth mentioning that each simulation is repeated 30 times in order to get accurate and trustful results.

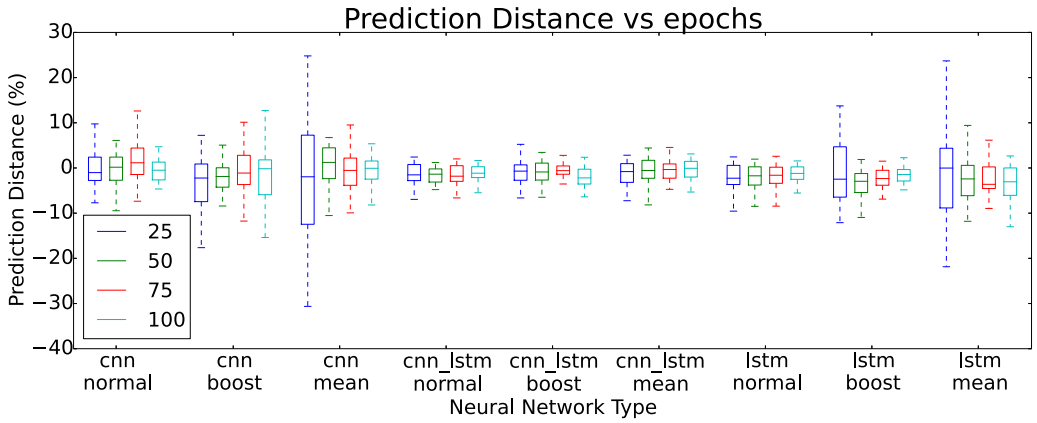
The simulations parameters are listed in the following:

- The number of clusters (equal to 30% of the size of training dataset when fixed)
- The number of epochs (equal to 100 epochs by training phase when fixed)
- The number of boost epochs (equal to 10 when fixed)

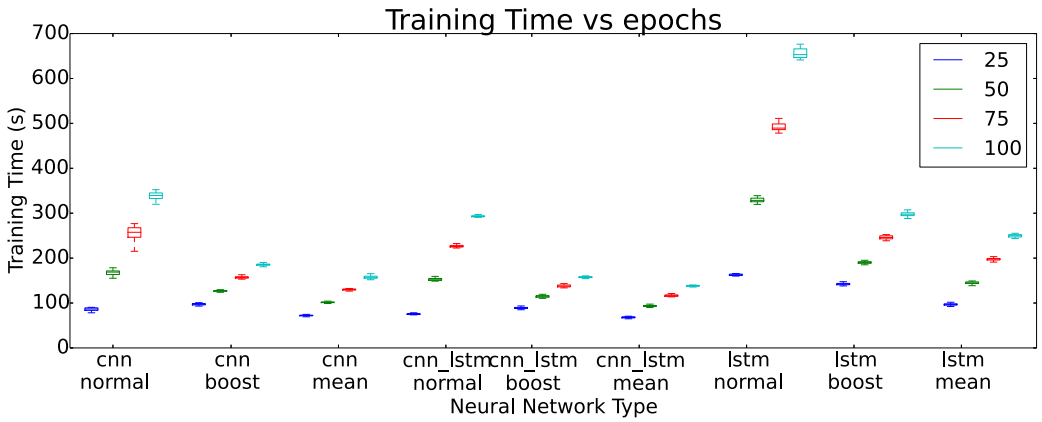
Finally, those metrics will be fixed in each simulation type where

5.6.3 Results and Analysis

From Figure 5.9 (a), we can see the effects of the number of epochs in the training phase for the different types of neural networks as with K-means clustering or not and with boost session or not. For all the simulations and the neural networks, it is obvious that 25 epochs is not enough to get precise prediction. On the other hand we can



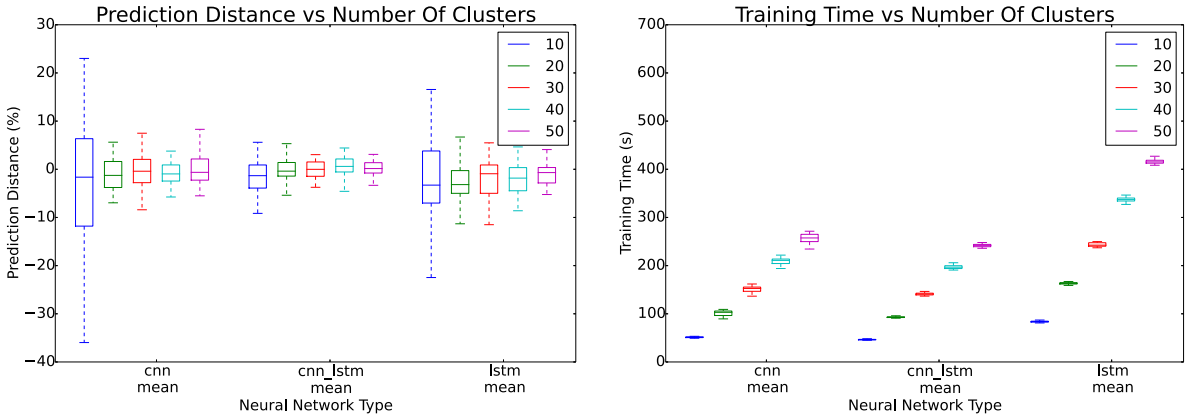
(a)



(b)

Figure 5.9: Evolution of prediction distance (a) and training phase duration (b) vs number of epochs in training phase

notice that using clustering technique for each neural network outputs approximately the same precision as using neural network with no data clustering. Thus, it shows that applying K-means clustering with randomly injection at each epoch allows to get the same precision. Also from Figure 5.9 (b), we can notice that the training time with K-means clustering is divided by two compared to the normal injection of the whole dataset. Therefore K-means clustering allows to reduce training duration while keeping the same precision as if we inject the whole training dataset. It is worth mentioning also that the boost at the end of an accelerated training phase is not so efficient as it does not add any precision to the output. So K-means clustering alone is a very interesting



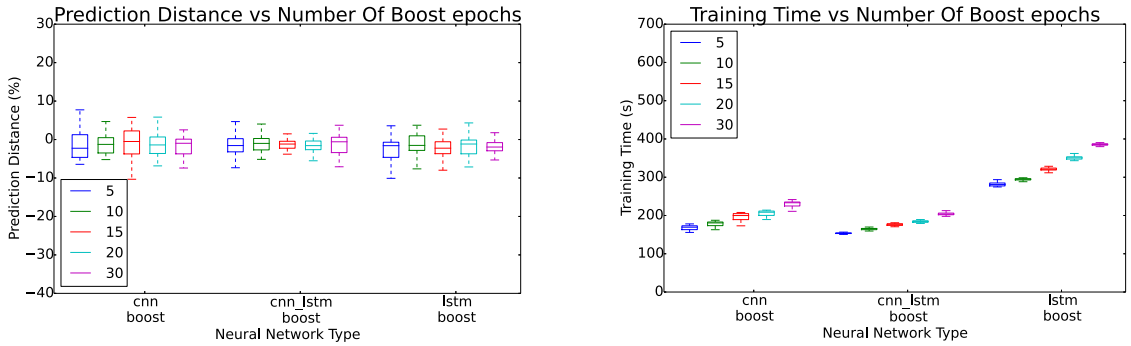
(a) (b)

Figure 5.10: Evolution of prediction distance (a) and training phase duration (b) vs number of clusters in training phase

solution in order to accelerate the training phase of the neural network.

In the same spirit Figure 5.10 (a) shows the effect of the variation of the number of clusters used to classify the training dataset, on the output precision. As illustrated in the Figure, if the number of clusters is around 10% of the training dataset size, the results are not that good. On the other hand from 20 % we start to get interesting results in term of output precision as the distance variation shrinks . Also figure 5.10 (b) shows the duration needed to train the neural network depending on the number of clusters. It shows that the duration increases with the number of clusters chosen, which is the expected behavior. But it is still worth mentioning that the duration needed to train the neural network even when the number of clusters is 50% of the training dataset size, is less than the training duration of a normal neural network with no generalization technique.

Figure 5.11 (a), shows the effect of adding some boost epochs at the end of an accelerated (using the proposed solution) training session. We can notice that the boost technique does not add precision. It even adds some confusion to the neural network as the prediction distances go beyond or beneath zero in some cases. In the same aspect boost epochs adds some latency to the duration of a training phase as it add more data for the last epochs. Thus boost epochs are not efficient in those use cases and using K-means clustering alone is better as it gives the desired accuracy while decreasing the training phase duration.



(a) (b)

Figure 5.11: Evolution of prediction distance (a) and training phase duration (b) vs number of boost in training phase

On the other hand from the Figures 5.9 (a), 5.10 (a) and 5.11 (a), we notice that the proposed model including CNN and LSTM is behaving very well and even better than the version of CNN standalone and the LSTM standalone. This is explained by the fact that the CNN allows extracting features from the input data and the LSTM allows classifying the output.

Summary

To sum up, applying K-means clustering with random injection at each epoch allows to get the same precision as injecting the whole training dataset, while dividing the training phase duration by two. In addition, the combined CNN-LSTM performs better than CNN standalone or LSTM standalone. Indeed, this architecture allows to benefit from CNN by extracting features and from LSTM output classification.

5.7 Conclusion

In this chapter we proposed: (i) a novel solution to scale dynamically and proactively the VNF elements of the 5G new generation mobile core network and (ii) a novel solution to generalize neural networks based on K-means clustering and Monte-Carlo method. The first technique is based on neural networks where two different models are compared: the DNN and the RNN. Simulations, based on a real dataset of a Network Operator, showed that our solution scales in/out accurately the number of

AMF instances depending on the load. Indeed, the proposed solution also bypasses the latency needed to provision new resources as it deploys in advance the needed number of AMF instances for the upcoming load; thus, avoiding overloading the core network and reducing the attach duration or the request process duration.

The second proposal uses real Operator's data in order to evaluate the efficiency of the proposed technique in predicting the upcoming network load for dynamic and proactive resource provisioning. The simulations showed that our solution allows reducing by 50% the duration of the training phase of a Neural Network. In addition, the proposed combined neural network showed clearly its efficiency in predicting the evolution of the considered time series as it performs better than a standalone CNN or a standalone LSTM.

CHAPTER 6

CONCLUSION

6.1 Introduction

With the emergence of 5G, new services and new types of connected devices will appear. Those services and connected devices have multiple requirements which lead to rise new challenges for the future 5G network. 5G is expected to serve services with ultra low latency, high bandwidth and massive connectivity requirements. In order to answer those needs by sharing the same physical infrastructures multiple key enablers should be envisioned in the development and conception of 5G. SDN, NFV and network slicing are key enablers allowing to answer those new requirements. Multiple studies and works have been conducted towards 5G. On the core network side, community research proposed multiple architectures introducing SDN and NFV. Some opted for the full Virtualisation of the control network as others decided to virtualise only the control plane. Also some have decided to integrate virtualised network functions while keeping some others as legacies one. Standardization groups also worked on the new generation core by introducing at a first stage a core network where the control and the data plane are separated. Then in a second phase they proposed the New Generation Core with modular functions benefiting from Virtualisation with new components and new communication interfaces.

In this thesis we focused on mobile core network performances and solutions for dynamic scalability of the components of the new generation core. As the number of connected devices tends to increase, dynamic scalability should be envisioned for the NGC. Performance benchmark was conducted and multiple algorithms were proposed in order to build a dynamic scaling mechanism.

6.2 Contributions of the thesis

The contributions of this thesis can be separated in three parts:

- **A benchmark of Mobile Core network architectures towards 5G:** In this contribution we have proposed a test-bed based on a real plateform using the openairinterface 5G project in order to emulate the RAN and openair-cn project to emulate the core network architectures. Three architectures were tested. The first one is based on the legacy 4G mobile core network but with virtualised components. The second one uses the virtualised 4G EPC but introduces an SDN controller in order to manage the data plane installation for the UEs. The third one is based on the 3GPP group architecture where the control plane and the data plane are separated. The results of this contribution showed that the new architectures based on SDN and NFV are as performant as the legacy 4G, rising hope for those key enablers to perform and offer the expected requirements of 5G. On the other side the benchmark allowed us to remark that a dynamic scaling mechanism is needed, especially for the MME components newly knew as AMF component in 5G. As this component is the single point of access to the core network, with the increase of the number of connected devices it would overload easily. Thus dynamic scalability should be addressed in the NGC.

- **Congestion control in NGC:** The second contribution tackles the scalability issue discovered in the first contribution by applying a congestion control mechanism. We developed a model of the core network procedures and thus designed a controller using control theory. This model allows to balance the load over the deployed AMF instances. At the same time, an algorithm was proposed also to scale in/out the instances dynamically by using the average load of the core network. This mechanism was accompanied by architectural proposal, where the algorithm should be run and also how statistical information could be retrieved in order to feed our algorithm based on 3GPP existing procedures. Matlab simulations were conducted in order to test our proposal and results were compared to an EWMA model. The obtained results clearly indicated that our proposal performs better than EWMA by dispatching the procedure arrival on the AMF as it is needed and by scaling in/ out the AMF instances depending on the need in a precise way.

- **Congestion avoidance in NGC:** Following the congestion control, we decided to propose a mechanism for congestion avoidance in NGC. Reacting to a congestion is important but it is more efficient if we are able to avoid it. With the emergence of new technologies and the development of machine Learning this is now possible. At first, in this contribution we proposed a way to format and to classify data to be used in order to predict the upcoming load of a mobile core network by using historical load data of an operator. We have designed and tested multiple types of neural networks in order to choose the best one. The DNN and the RNN (LSTM cell) have both predicted well the upcoming load based on a historical dataset but the RNN was more precise comparing to reality. In order to push more the tests, we developed a Discrete event simulator in order to simulate the behavior of the NGC and thus test our proposal, which scales in advance the AMF components based on the prediction of the Neural Network. The results have shown that the neural network based solution behaves a lot better than a static threshold based solution. It deploys exactly the needed number of AMF instances and avoids deployment latency as it anticipates the increase of the load. To push further this contribution we decided to explore more the potential of the Neural Network technology by generalizing the used Neural Network and thus allow adaptation to traffic pattern change. In this part we decided to use a CNN neural network combined to LSTM cell. This allows to benefit from the potential of CNN to extract features and the potential of LSTM that is classifying the output. Also in this part we have worked on accelerating the training phase of a neural network by classifying the data using K-mean procedures. Indeed this proposal allowed to reduce by 50% the duration of the training phase and to allow the neural network to adapt to pattern change without a new training phase.

6.3 Perspectives

To extend the work of the thesis, the following research axes represent an interesting topics to explore:

- **Horizontal scaling in the NGC:** In the context of the thesis we worked on scaling and load balancing the AMF instance of the NGC as it is the main entry point of the network, while it presents the more chances to be congested with the increase

of the number of connected devices. Also this proposal serves to save resources allocation as we deploy only what is needed to absorb the actual load. On the other hand in the NGC context, multiple SMF components could be deployed and also each one could be dedicated for a certain type of application. Thus, the dynamic scaling mechanism should be extended in order to take into account the number of SMF instances connected to an AMF. Hence, the load balancing procedure will take into account more parameters that will allow to balance the load not only on the access side of the core network but also deeply in the core network.

- **Micro Services in NGC for 5G:** The architecture of the NGC is not completely defined for 5G. The way to deploy the core network architecture proposed for 5G should be revised also in order to completely benefit from the existing key enablers and thus gain more in elasticity and flexibility. One of the key enablers that should be investigated is micro services for NGC. Micro services will allow to create pools by function and to break monolithic blocks. Indeed it will make the Core network functions smaller, more fined and thus easier to understand, to develop and to evolve [103]. In addition, as we are in a virtualised and cloud environment, micro services allow fault isolation where larger functions could remain unaffected by a single faulty module.

- **More specialized neural networks for 5G load prediction:** In the context of the thesis we have proposed to use neural networks in order to predict the upcoming load in a mobile core network based on a historical dataset. We also proposed a generalization of the neural network in order to adapt for pattern changes in the upcoming load. Therefore, in our proposal we are able to detect and adapt for an amplitude increase or decrease. So, a neural network able to detect the shape or the pattern of the arrival function and to adapt for shape changes is needed. Thus, the neural network will be more accurate and stronger in adaptation. Also a mechanism able to compare the real and the predicted load in order to correct the gap in case of prediction error and re-adapt in order to serve as efficient as possible the arrivals. For more details on Neural networks in mobile networking, readers may refer to [104].

PUBLICATIONS FROM THE THESIS

1. ***Alawe, I.***, Ksentini, A., Hadjadj-Aoul, Y., Bertin, P., & Kerbellec, A. (2017, September). On evaluating different trends for virtualized and SDN-ready mobile network. In Cloud Networking (CloudNet), 2017 IEEE 6th International Conference on (pp. 1-6). IEEE.

2. ***Alawe, I.***, Hadjadj-Aoul, Y., Bertin, P., Ksentini, A., & Darche, D. (2017, November). RNN-based traffic prediction for pro-active scaling of the AMF. In SDN Day 2017-Software Defined Networks Day.

3. ***Alawe, I.***, Hadjadj-Aoul, Y., Ksentini, A., Bertin, P., & Darche, D. (2018, January). On the scalability of 5G Core network: the AMF case. In Consumer Communications & Networking Conference (CCNC), 2018 15th IEEE Annual (pp. 1-6). IEEE.

4. ***Alawe, I.***, Hadjadj-Aoul, Y., Ksentini, A., Bertin, P., Viho, C., & Darche, D. (2018, October). An efficient and lightweight load forecasting for proactive scaling in 5G mobile networks. In Conference on Standards for Communications and Networking (CSCN), 2018, IEEE.

5. ***Alawe, I.***, Ksentini, A., Hadjadj-Aoul, Y., & Bertin, P. (2018, November). Improving traffic forecasting for 5G core network scalability: A Machine Learning approach. In IEEE Network Magazine.

6. ***Alawe, I.***, Hadjadj-Aoul, Y., Ksentini, A., Bertin, P., Viho, C., & Darche, D. (2018, December). Smart scaling of the 5G core network: an RNN-based approach. In Global Communications Conference (GLOBECOM), 2018, IEEE.

BIBLIOGRAPHY

[1] A. Nordrum, “Popular internet of things forecast of 50 billion devices by 2020 is outdated,” IEEE Spectrum. [Online]. Available: https://goo.gl/5TS9s5

[2] “3rd generation partnership project (3GPP).” [Online]. Available: http://www.3gpp.org

[3] “NGMN, 5g white paper.” [Online]. Available: https://www.ngmn.org/5g-white-paper.html

[4] *Study on control and user plane seperation of EPC nodes*, 3GPP, 06 2016.

[5] 3rd Generation Partnership Project (3GPP), “GPRS Tunneling Protocol.” [Online]. Available: http://www.arib.or.jp/IMT-2000/V720Mar09/5_Appendix/Rel8/29/29281-800.pdf

[6] OpenNetworkingFoundation, “OpenFlow switch specification.”

[7] “OpenAirInterface5G.” [Online]. Available: https://gitlab.eurecom.fr/oai/openairinterface5g

[8] “OpenAirCoreNetwork.” [Online]. Available: https://gitlab.eurecom.fr/oai/openair-cn

[9] “OpenDayLight.” [Online]. Available: https://www.opendaylight.org/

[10] “OpenvSwitch.” [Online]. Available: http://openvswitch.org/

[11] 3rd Generation Partnership Project (3GPP), “System architecture for the 5g system.” [Online]. Available: http://www.3gpp.org/ftp/specs/archive/23_series/23.501/

[12] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, P. Bertin, and A. Kerbellec, “On evaluating different trends for virtualized and sdn-ready mobile network,” in *Cloud Networking (CloudNet), 2017 IEEE 6th International Conference on*. IEEE, 2017, pp. 1–6.

[13] I. Alawe, Y. Hadjadj-Aoul, A. Ksentini, P. Bertin, and D. Darche, “On the scalability of 5g core network: the amf case,” in *IEEE Consumer Communications and Networking Conference (CCNC)*, 2018.

[14] I. Alawe, Y. Hadjadj-Aoul, P. Bertin, A. Ksentini, and D. Darche, “Rnn-based traffic prediction for pro-active scaling of the amf,” in *SDN Day 2017-Software Defined Networks Day*, 2017, p. 1.

[15] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, “Improving traffic forecasting for 5g core network scalability: A machine learning approach,” *IEEE network magazine*, 2018.

[16] I. Alawe, Y. Hadjadj-Aoul, A. Ksentini, P. Bertin, C. Viho, and D. Darche, “Smart scaling of the 5g core network: an rnn-based approach,” in *Global Conference (Globecom)*. IEEE, 2018.

[17] ——, “An efficient and lightweight load forecasting for proactive scaling in 5g mobile networks,” in *Conference on Standards for Communications and Networking (CSCN)*. IEEE, 2018.

[18] S. Li, L. Da Xu, and S. Zhao, “5g internet of things: A survey,” *Journal of Industrial Information Integration*, 2018.

[19] IEEE, “Ieee 5g standards.” [Online]. Available: https://5g.ieee.org/standards

[20] B. Blanco, J. O. Fajardo, I. Giannoulakis, E. Kafetzakis, S. Peng, J. Pérez-Romero, I. Trajkovska, P. S. Khodashenas, L. Goratti, M. Paolino *et al.*, “Technology pillars in the architecture of future 5g mobile networks: Nfv, mec and sdn,” *Computer Standards & Interfaces*, vol. 54, pp. 216–228, 2017.

[21] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, “Software-defined networking: A comprehensive survey,” *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[22] H. Farhady, H. Lee, and A. Nakao, “Software-defined networking: A survey,” *Computer Networks*, vol. 81, pp. 79–95, 2015.

[23] Sdxcentral, “What is nfv – network functions virtualization – definition?” [Online]. Available: https://www.sdxcentral.com/nfv/definitions/whats-network-functions-virtualization-nfv/

[24] *Network Functions Virtualisation (NFV); Management and Orchestration*. The European Telecommunications Standards Institute, December 2014.

[25] T. Yoo, "Network slicing architecture for 5g network," in *Information and Communication Technology Convergence (ICTC), 2016 International Conference on*. IEEE, 2016, pp. 1010–1014.

[26] 3GPP, "Telecommunication management;study on management and orchestration of network slicing for next generation network." [Online]. Available: https://bit.ly/2qeZd1s

[27] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5g: Survey and challenges," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 94–100, 2017.

[28] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing & softwarization: A survey on principles, enabling technologies & solutions," *IEEE Communications Surveys & Tutorials*, 2018.

[29] 3rd Generation Partnership Project (3GPP), "Network architecture." [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=728

[30] G. M. Frédéric Firmin, "The evolved packet core." [Online]. Available: http://www.3gpp.org/technologies/keywords-acronyms/100-the-evolved-packet-core

[31] A. Basta, W. Kellerer, M. Hoffmann, K. Hoffmann, and E.-D. Schmidt, "A virtual sdn-enabled lte epc architecture: a case study for s-/p-gateways functions," in *Future Networks and Services (SDN4FNS)*. IEEE, 2013, pp. 1–7.

[32] A. Basta, A. Blenk, M. Hoffmann, H. J. Morper, K. Hoffmann, and W. Kellerer, "Sdn and nfv dynamic operation of lte epc gateways for time-varying traffic patterns," in *International Conference on Mobile Networks and Management*. Springer, 2014, pp. 63–76.

[33] A. Basta, W. Kellerer, M. Hoffmann, H. J. Morper, and K. Hoffmann, "Applying nfv and sdn to lte mobile core gateways, the functions placement problem," in *Proceedings of the 4th workshop on All things cellular: operations, applications, & challenges*. ACM, 2014, pp. 33–38.

[34] J. Mwangama and N. Ventura, “Accelerated virtual switching support of 5g nfv-based mobile networks,” in *Personal, Indoor, and Mobile Radio Communications (PIMRC), 2017 IEEE 28th Annual International Symposium on*. IEEE, 2017, pp. 1–7.

[35] “Intel data plane development kit.” [Online]. Available: https://www.dpdk.org/

[36] X. Jin, L. E. Li, L. Vanbever, and J. Rexford, “Softcell: Scalable and flexible cellular core network architecture,” in *Proceedings of the ninth ACM conference on Emerging networking experiments and technologies*. ACM, 2013, pp. 163–174.

[37] M. Moradi, L. E. Li, and Z. M. Mao, “Softmow: A dynamic and scalable software defined architecture for cellular wans,” in *Proceedings of the third workshop on Hot topics in software defined networking*. ACM, 2014, pp. 201–202.

[38] L. E. Li, Z. M. Mao, and J. Rexford, “Toward software-defined cellular networks,” in *Software Defined Networking (EWSDN), 2012 European Workshop on*. IEEE, 2012, pp. 7–12.

[39] J. Costa-Requena, “Sdn integration in lte mobile backhaul networks,” in *Information Networking (ICOIN), 2014 International Conference on*. IEEE, 2014, pp. 264–269.

[40] A. C. Morales, A. Aijaz, and T. Mahmoodi, “Taming mobility management functions in 5g: handover functionality as a service (faas),” in *Globecom Workshops (GC Wkshps), 2015 IEEE*. IEEE, 2015, pp. 1–4.

[41] T. Mahmoodi and S. Seetharaman, “On using a sdn-based control plane in 5g mobile networks,” in *Wireless World Research Forum, 32nd Meeting*. Citeseer, 2014.

[42] S. Chourasia and K. M. Sivalingam, “Sdn based evolved packet core architecture for efficient user mobility support,” in *Network Softwarization (NetSoft), 2015 1st IEEE Conference on*. IEEE, 2015, pp. 1–5.

[43] A. Bradai, K. Singh, T. Ahmed, and T. Rasheed, “Cellular software defined networking: a framework,” *IEEE communications magazine*, vol. 53, no. 6, pp. 36–43, 2015.

[44] S. Shanmugalingam and P. Bertin, “Programmable mobile core network,” in *ISCC*. IEEE, 2014, pp. 1–7.

[45] H. Ali-Ahmad, C. Cicconetti, A. De la Oliva, V. Mancuso, M. R. Sama, P. Seite, and S. Shanmugalingam, “An sdn-based network architecture for extremely dense wireless networks,” in *Future Networks and Services (SDN4FNS), 2013 IEEE SDN for*. IEEE, 2013, pp. 1–7.

[46] H. Ali-Ahmad, C. Cicconetti, A. de la Oliva, M. Dräxler, R. Gupta, V. Mancuso, L. Roullet, and V. Sciancalepore, “Crowd: an sdn approach for densenets,” 2013.

[47] M. H. Kabir, “A novel architecture for sdn-based cellular network,” *International Journal of Wireless & Mobile Networks*, vol. 6, no. 6, p. 71, 2014.

[48] J. Kempf, B. Johansson, S. Pettersson, H. Lüning, and T. Nilsson, “Moving the mobile evolved packet core to the cloud,” in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on*. IEEE, 2012, pp. 784–791.

[49] J. Mueller, Y. Chen, B. Reichel, V. Vlad, and T. Magedanz, “Design and implementation of a carrier grade software defined telecommunication switch and controller,” in *NOMS*. IEEE, 2014, pp. 1–7.

[50] V.-G. Nguyen and Y. Kim, “Signaling load analysis in openflow-enabled lte/epc architecture,” in *Information and Communication Technology Convergence (ICTC), 2014 International Conference on*. IEEE, 2014, pp. 734–735.

[51] ——, “Proposal and evaluation of sdn-based mobile packet core networks,” *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, pp. 1–18, 2015.

[52] G. Hampel, M. Steiner, and T. Bu, “Applying software-defined networking to the telecom domain,” in *Computer Communications Workshops (INFOCOM WKSHPS), 2013 IEEE Conference on*. IEEE, 2013, pp. 133–138.

[53] K. Pentikousis, Y. Wang, and W. Hu, “Mobileflow: Toward software-defined mobile networks,” *IEEE Communications magazine*, vol. 51, no. 7, pp. 44–53, 2013.

[54] J. Macker, “Simplified multicast forwarding,” Tech. Rep., 2012.

[55] S. Matsushima and R. Wakikawa, “Stateless user-plane architecture for virtualized epc (vepc),” *draft-matsushima-stateless-uplane-vepc-04 (work in progress)*, 2015.

[56] S. Ben Hadj Said, M. R. Sama, K. Guillouard, L. Suciu, G. Simon, X. Lagrange, and J.-M. Bonnin, “New control plane in 3gpp lte/epc architecture for on-demand connectivity service,” in *CloudNet*. IEEE, 2013, pp. 205–209.

[57] M. R. Sama, S. B. H. Said, K. Guillouard, and L. Suciu, “Enabling network programmability in lte/epc architecture using openflow,” in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2014 12th International Symposium on*. IEEE, 2014, pp. 389–396.

[58] X. An, W. Kiess, and D. Perez-Caparros, “Virtualization of cellular network epc gateways based on a scalable sdn architecture,” in *GLOBECOM*. IEEE, 2014, pp. 2295–2301.

[59] V.-G. Nguyen, T.-X. Do, and Y. Kim, “Sdn and virtualization-based lte mobile network architectures: A comprehensive survey,” *Wireless Personal Communications*, vol. 86, no. 3, pp. 1401–1438, 2016.

[60] V.-G. Nguyen, A. Brunstrom, K.-J. Grinnemo, and J. Taheri, “Sdn/nfv-based mobile packet core network architectures: A survey,” *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.

[61] Z. Zaidi, V. Friderikos, Z. Yousaf, S. Fletcher, M. Dohler, and H. Aghvami, “Will sdn be part of 5g?” *IEEE Communications Surveys & Tutorials*, 2018.

[62] A. Ksentini, Y. Hadjadj-Aoul, and T. Taleb, “Cellular-based machine-to-machine: overload control,” *IEEE Network*, vol. 26, no. 6, pp. 54–60, November 2012.

[63] A. Ksentini, M. Bagaa, and T. Taleb, “On using sdn in 5g: the controller placement problem,” in *Global Conference (Globecom)*. IEEE, 2016.

[64] D. Nowoswiat, “Managing lte core network signaling traffic.” [Online]. Available: https://goo.gl/2pjygw

[65] A. S. Rajan and K. B. Ramia, “Application of nfv and sdn to 5g infrastructure,” *Towards 5G: Applications, Requirements and Candidate Technologies*, pp. 408–420, 2016.

[66] H. Hawilo, A. Shami, M. Mirahmadi, and R. Asal, “Nfv: state of the art, challenges, and implementation in next generation mobile networks (vepc),” *IEEE Network*, vol. 28, no. 6, pp. 18–26, 2014.

[67] T. Phung-Duc, Y. Ren, J.-C. Chen, and Z.-W. Yu, “Design and analysis of deadline and budget constrained autoscaling (dbca) algorithm for 5g mobile networks,” *arXiv preprint arXiv:1609.09368*, 2016.

[68] Y. Ren, T. Phung-Duc, J.-C. Chen, and Z.-W. Yu, “Dynamic auto scaling algorithm (dasa) for 5g mobile networks,” in *GLOBECOM*. IEEE, 2016.

[69] S. Jeon, D. Corujo, and R. L. Aguiar, “Virtualised epc for on-demand mobile traffic offloading in 5g environments,” in *CSCN*. IEEE, 2015.

[70] Y. Takano, A. Khan, M. Tamura, S. Iwashina, and T. Shimizu, “Virtualization-based scaling methods for stateful cellular network nodes using elastic core architecture,” in *CloudCom*. IEEE, 2014.

[71] X. An, F. Pianese, I. Widjaja, and U. G. Acer, “Dmme: A distributed lte mobility management entity,” *Bell Labs Technical Journal*, vol. 17, no. 2, pp. 97–120, 2012.

[72] A. Banerjee, R. Mahindra, K. Sundaresan, S. Kasera, K. Van der Merwe, and S. Rangarajan, “Scaling the lte control-plane for future mobile access,” in *CoNEXT*. ACM, 2015, p. 19.

[73] J. Prados-Garzon, J. J. Ramos-Munoz, P. Ameigeiras, P. Andres-Maldonado, and J. M. Lopez-Soler, “Modeling and dimensioning of a virtualized mme for 5g mobile networks,” *arXiv preprint arXiv:1703.04445*, 2017.

[74] G. Premsankar, K. Ahokas, and S. Luukkainen, “Design and implementation of a distributed mobility management entity on openstack,” in *CloudCom*. IEEE, 2015, pp. 487–490.

[75] ETSI and 3GPP, “Ts 136 413.” [Online]. Available: https://goo.gl/CK7qSG

[76] IETF, “DNS Push Notifications.” [Online]. Available: https://goo.gl/GGuWeC

[77] M. C. de Oliveira, J. Bernussou, and J. C. Geromel, “A new discrete-time robust stability condition,” *Systems & control letters*, vol. 37, no. 4, pp. 261–265, 1999.

[78] R. E. Kalman, “Contributions to the theory of optimal control,” 1960.

[79] K. Zhou, J. C. Doyle, K. Glover *et al.*, *Robust and optimal control*. Prentice hall New Jersey, 1996, vol. 40.

[80] C. Wang, B. Li, Y. T. Hou, K. Sohraby, and Y. Lin, “Lred: a robust active queue management scheme based on packet loss ratio,” in *INFOCOM 2004*, vol. 1. IEEE, 2004.

[81] S. Dutta, T. Taleb, and A. Ksentini, “Qoe-aware elasticity support in cloud-native 5g systems,” in *ICC*. IEEE, 2016, pp. 1–6.

[82] G. A. Carella, M. Pauls, L. Grebe, and T. Magedanz, “An extensible autoscaling engine (ae) for software-based network functions,” in *NFV-SDN*. IEEE, 2016, pp. 219–225.

[83] A. B. Alvi, T. Masood, and U. Mehboob, “Load based automatic scaling in virtual ip based multimedia subsystem,” in *CCNC*. IEEE, 2017, pp. 665–670.

[84] Amazon, “Web services auto scaling.” [Online]. Available: https://aws.amazon.com/autoscaling/

[85] A. CloudStack, “Open source cloud computing.” [Online]. Available: http://cloudstack.apache.org/

[86] C. H. T. Arteaga, F. Rissoi, and O. M. C. Rendon, “An adaptive scaling mechanism for managing performance variations in network functions virtualization: A case study in an nfv-based epc,” in *2017 13th International Conference on Network and Service Management (CNSM)*. IEEE, 2017, pp. 1–7.

[87] A. Bilal, T. Tarik, A. Vajda, and B. Miloud, “Dynamic cloud resource scheduling in virtualized 5g mobile systems,” in *Global Communications Conference (GLOBECOM), 2016 IEEE*. IEEE, 2016, pp. 1–6.

[88] R. Fu, Z. Zhang, and L. Li, “Using lstm and gru neural network methods for traffic flow prediction,” in *Chinese Association of Automation (YAC), Youth Academic Annual Conference of*. IEEE, 2016, pp. 324–328.

[89] K. M. Faraoun and A. Boukelif, “Neural networks learning improvement using the k-means clustering algorithm to detect network intrusions,” *International Journal of Computational Intelligence*, vol. 5, no. 3, pp. 161–168, 2007.

[90] T. Italia, “Telecom italia big data challenge.” [Online]. Available: http://www.telecomitalia.com/tit/en/bigdatachallenge/contest.html

[91] ——, “A multi-source dataset of urban life in the city of milan and the province of trentino dataverse.” [Online]. Available: https://dataverse.harvard.edu/dataset.xhtml?persistentId=doi:10.7910/DVN/EGZHFV

[92] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[93] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” in *Interspeech*, 2014.

[94] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*. MIT press Cambridge, 2016, vol. 1.

[95] Y. LeCun, Y. Bengio *et al.*, “Convolutional networks for images, speech, and time series,” *The handbook of brain theory and neural networks*, vol. 3361, no. 10, p. 1995, 1995.

[96] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. [Online]. Available: https://doi.org/10.1162/neco.1997.9.8.1735

[97] TensorFlow, “An open-source machine learning framework for everyone.” [Online]. Available: https://www.tensorflow.org/

[98] Team-SimPy, “Discrete event simulation for python.” [Online]. Available: https://simpy.readthedocs.io/en/latest/

[99] P. A. Frangoudis, L. Yala, and A. Ksentini, “Cdn-as-a-service provision over a telecom operator’s cloud,” *IEEE Transactions on Network and Service Management*, vol. 14, no. 3, pp. 702–716, 2017.

[100] S. Basterrech, G. Rubino, and V. Snasel, “Sensitivity analysis of echo state networks for forecasting pseudo-periodic time series,” in *2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR)*, Nov 2015, pp. 328–333.

[101] F. Gers, D. Eck, and J. Schmidhuber, “Applying lstm to time series predictable through time-window approaches,” *Springer Perspectives in Neural Computing*, vol. Neural Nets WIRN Vietri-01, pp. 193–20, 2002.

[102] E. J. Keogh and M. J. Pazzani, “Scaling up dynamic time warping for datamining applications,” in *Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD ’00. New York, NY, USA: ACM, 2000, pp. 285–289.

[103] CISCO, “Evolving the mobile core to being cloud native.” [Online]. Available: https://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/ultra-services-platform/white-paper-c11-739215.pdf

[104] C. Zhang, P. Patras, and H. Haddadi, “Deep learning in mobile and wireless networking: A survey,” *arXiv preprint arXiv:1803.04311*, 2018.

ACRONYMS

3GPP 3rd Generation Partnership Project.

4G 4th Generation Of Mobile Networks.

5G 5th Generation Of Mobile Networks.

AMF Access and Mobility Function.

API Application Programming Interface.

BGP Border Gateway Protocol.

CapEx Capital Expenditure.

CDF Cumulative Distribution Function.

CN Core Network.

CNF Core Network Function.

CNN Convolutional Neural Network.

CUPS Control and User Plane Separation.

DNN Deep Neural Network.

DNS Domain Name System.

DPDK Data Plane Development Kit.

DTW Dynamic Time Warping.

EM Element Manager.

EPC Evolved Packet Core.

ETSI European Telecommunications Standards Institute.

EUTRAN Evolved Universal Terrestrial Radio Access Network.

evEPC Enhanced Virtual EPC.

EWMA Exponentially Weighted Moving Average.

FFNN Feed Forward Neural Network.

GRE Generic Routing Encapsulation.

GTP GPRS Tunneling Protocol.

HSS Home Subscriber Server.

IMSI International Mobile Subscriber Identities.

IOT Internet Of Things.

KPI Key Performance Indicator.

LQR Linear Quadratic Regulator.

LSTM Long Short Term Memory.

LTE Long Term Evolution.

MAC Medium Access Control.

MANO Management And Orchestration.

MME Mobility and Management Entity.

MNO Mobile Network Operators.

MTC Machine Type Communication.

NAS Non-Access Stratum.

NF Network Functions.

NFV Network Functions Virtualisation.

NFVI Network Functions Virtualisation Infrastructure.

NFVM Network Functions Virtualisation Manager.

NFVO NFV Orchestrator.

NGC New Generation Core.

NGMN Next Generation Mobile Networks.

NOS Network Operating System.

NRF NF Repository Function.

NS Network Services.

ODL OpenDayLight.

OF OpenFlow Protocol.

ONF Open Networking Group.

OpEx Operational Expenditure.

OVS Open Virtual Switch.

PCF Policy Control Function.

PCRF Policy and Charging Rules Function.

PDCP Packet Data Convergence Protocol.

PDN Packet Data Network.

PGW PDN GateWay.

PGW-U PDN GateWay User Plane functions.

PGW-C PDN GateWay Control Plane functions.

QoS Quality of Service.

RAN Radio Access Network.

ReLU Rectified Linear Unit.

REST Representational State Transfer.

RNN Recurrent Neural Network.

RRC Radio Resource Control.

RTT Round-Trip Time.

S/PGW Serving GateWay/PDN GateWay User Plane functions.

S/PGW-U Serving GateWay/PDN GateWay User Plane functions.

S/PGW-C Serving GateWay/PDN GateWay Control Plane functions.

SDN Software Defined Networking.

SGW Serving GateWay.

SGW-U Serving GateWay User Plane functions.

SGW-C Serving GateWay Control Plane functions.

SMF Session Management Function.

SNMP Simple Network Management Protocol.

SO Service Orchestrator.

UDM Unified Data Management.

UE User Equipment.

UPF User Plane Function.

vEPC Virtual EPC.

VIM Virtualised Infrastructure Manager.

VM Virtual Machine.

VNF Virtual Network Functions.

VNFM VNF Manager.

XDP eXpress Data Path.

UNIVERSITE BRETAGNE LOIRE / MATHSTIC

UNIVERSITÉ DE RENNES 1

Titre: Évaluation d'architectures et scalabilité dynamique des réseaux mobiles cœur 5G

Mot clés : Réseaux Mobile Cœur, 5G, Prédiction de trafic, Scalabilité

Resumé : Afin de répondre aux besoins de la 5g, nous évaluons plusieurs visions du cœur de réseau. Nous comparons les performances des visions en mesurant le temps nécessaire pour établir le service pour l'utilisateur. De plus, nous proposons dans cette thèse un algorithme basé sur la théorie du contrôle permettant d'équilibrer la charge entre les instances AMF, et d'augmenter ou de diminuer le nombre d'instances AMF en fonction de la charge du réseau. En outre, nous proposons un nouveau mécanisme pour adapter les ressources du réseau cœur 5G en anticipant les évolutions, de la charge de trafic, grâce à des prédictions via des approches de machine learning. Enfin, nous proposons une solution pour généraliser les réseaux de neurones tout en accélérant le processus.

Title: Architectures evaluation and dynamic scaling for 5G mobile core networks

Keywords : Mobile core networks, 5G, Traffic prediction, Scalability

Abstract : In order to fulfill the needs of 5G, we evaluate, using a testbed, multiple visions of core networks. We compare the performances of the visions in terms of the time needed to create the user data plane. In addition, we provide an algorithm based on Control Theory allowing to equilibrate the load on the AMF instances, and to scale out or in the AMF instances depending on the network load. Also, we propose a novel mechanism to scale 5G core network resources by anticipating traffic load changes through forecasting via deep learning approaches. Finally, we propose a novel solution to generalize neural networks while accelerating the learning process.