ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

SPATIO-TEMPORAL DATA PLANE DESIGN FOR SOFTWARE DEFINED CELLULAR NETWORKS (SDcN)

M.Sc. THESIS

Yusuf ÖZÇEVİK

Department of Computer Engineering

Computer Engineering Programme

MAY 2015

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

SPATIO-TEMPORAL DATA PLANE DESIGN FOR SOFTWARE DEFINED CELLULAR NETWORKS (SDcN)

M.Sc. THESIS

Yusuf ÖZÇEVİK
(504131549)

Department of Computer Engineering

Computer Engineering Programme

Thesis Advisor: Asst. Prof. Dr. Berk CANBERK

MAY 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

YAZILIM TABANLI HÜCRESEL AĞLAR (YThA) İÇİN UZAYSAL-ZAMANSAL VERİ KATMANI TASARIMI

YÜKSEK LİSANS TEZİ

Yusuf ÖZÇEVİK
(504131549)

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Tez Danışmanı: Asst. Prof. Dr. Berk CANBERK

MAYIS 2015

Yusuf ÖZÇEVİK, a M.Sc. student of ITU Graduate School of Science Engineering and Technology 504131549 successfully defended the thesis entitled **"SPATIO-TEMPORAL DATA PLANE DESIGN FOR SOFTWARE DEFINED CELLULAR NETWORKS (SDcN)"**, which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Asst. Prof. Dr. Berk CANBERK** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Özgür Barış Akan** Koç University	.....
	Assoc. Prof. Dr. Güneş Zeynep Karabulut Kurt Istanbul Technical University	.....
		.....

Date of Submission : **4 May 2015**
Date of Defense : **27 May 2015**

To my fiance and family,

FOREWORD

In my masters degree period, I have worked on the novel SDcN fashion. In my thesis, I focus on an innovative swtich architecture desing for Data Plane part of this technology in order to improve QoS related performance issues. During that progress, there are some valuable people that encourage me on my work and also on my master education. I take this opportunity to express my gratitude to all of these people.

During this thesis, my masters degree advisor and also my bachelors degree advisor, Asst. Prof. Dr. Berk Canberk has been an excellent advisor for me. His ideas and comments about my works always encouraged me to work eagerly since my very first projects in my BSc. I would like to send my special thanks to him not only for being my advisor; but also a significant person for my career. He encouraged and inspired me to start my masters education in my university. He always had a pretty understanding about my professional job and facilitated works for my masters degree when I had some busy periods with other jobs.

I would like to thank to my family including my dad, my mum, my sister and especially my fiance Müge. My family always supported me even we met only few times per year because of physical geography that we lived. I always felt their support with me in troublesome situations and always celebrated my achievements with them. Müge is the largest shareholder in all my achievements. She have always inspired me for five years and I did everything for her, for a life with her. We involved many works together and achieved great successes that encourage both of us. Her father and mother, my future father in law and mother in law, also supported me in every step that I took. They become a part of my great family and never make me feel alone in this city. Moreover, I am also grateful to my close friends Yiğit, Fahrettin and Osman. I spent much time with them in my masters degree, they always supported me for all my works.

I am grateful to all my colleagues working in Computing Department of Istanbul Technical University, my director and also my teacher A. Cüneyd Tantuğ and my group director Asım Güneş. My colleagues were always good to me and my directors always had a great understanding to allocate time for my masters degree courses and works.

Lastly, I also place on record, my sense of gratitude to one and all, who directly or indirectly, have lemt their hand in my work.

May 2015

Yusuf ÖZÇEVİK
(Computer Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

SDcN : Software Defined Cellular Network
QoS : Quality of Service
OF : OpenFlow
MsOF : Multi-stage OpenFlow
NoC : Network on Chip
w.r.t. : with respect to
YThA : Yazılım Tanımlı Hücresel Ağ

LIST OF FIGURES

Page

LIST OF SYMBOLS

N	:	Number of Input and Output Ports in a Switch
C	:	Spatial Complexity Parameter
T	:	Temporal Complexity Parameter
C_{MsOF}	:	Spatial Complexity Parameter of Multi-stage OF Switch
C_{CON}	:	Spatial Complexity Parameter of Conventional OF Switch
T_{MsOF}	:	Temporal Complexity Parameter of Multi-stage OF Switch
T_{CON}	:	Temporal Complexity Parameter of Conventional OF Switch
n	:	Number of Input Ports for each Flow Tables in $Stage_1$ and also Number of Output Ports for each Flow Tables in $Stage_3$ of an MsOF Switch
k	:	Number of Flow Tables in $Stage_2$ of MsOF Switch
b_{CON}	:	Number of Bits must be Matched in MsOF Switch Flow Table
b_{MsOF}	:	Number of Bits must be Matched in Conventional OF Switch Flow Table
t_r	:	Delay for Exact Matching in a Row of Flow Table
t_{MsOF}	:	Processing Time of a Flow at each Micropocessor in Multi-stage OF Switch
$\lambda^{[x][y]}$	:	Arrival Rate of Flows in x^{th} Stage and y^{th} Pipeline
$N^{[x][y]}$	:	Expected Number of Flows in x^{th} Stage and y^{th} Pipeline
$\lambda^{'}$	:	Total Arrival Rate of Flows in the System
$\lambda^{'}_j$	:	Total Arrival Rate of Flows in the System at j^{th} OF Switch
$\rho_j(t)$	:	Utilization of M/M/1 System w.r.t. Time t
μ_{CON1}	:	Service Rate of a Table for Search Operation
μ_{CON2}	:	Service Rate of the Microprocessor at the end of Conventional Pipeline
μ_{MsOF}	:	Total Service Rate of Table and Microprocessor in each Multi-stage M/M/1 System
N_j^{1st}	:	Expected Number of Flows in a Table of Conventional M/M/1 System at j^{th} OF Switch
N_j^{2nd}	:	Expected Number of Flows in Microprocessor of Conventional M/M/1 System at j^{th} OF Switch
$N^{'}_j$	:	Total Expected Number of Flows in j^{th} OF Switch
t_{CON2}	:	Service Time of the Microprocessor at the end of Conventional Pipeline for Flow Processing
$T^{'}_j$	:	Total Time Spent of a Flow in j^{th} OF Switch
p_{1to2}	:	Propobability of a Flow to Going from $Stage_1$ to $Stage_2$ for each Pipeline
p_{2to3}	:	Propobability of a Flow to Going from $Stage_2$ to $Stage_3$ for each Pipeline
i, j, x, y, k	:	Index Symbols to Indicate a Specific Entity in the System

SPATIO-TEMPORAL DATA PLANE DESIGN FOR SOFTWARE DEFINED CELLULAR NETWORKS (SDcN)

SUMMARY

The tremendous increase on mobile data traffic has stressed conventional cellular networks recently. Network management has also become difficult because of enormous traffic demand. At this point, a flexible and dynamic control mechanism is needed that diminishes the complexity in physical topology. SDcN, is one of the novel next generation approaches that makes it easier to orchestrate physical devices in Data Plane with its centralized control fashion. SDcN paradigm provides a simple management strategy for network equipments in physical topology by moving their control logic to Control Plane. These equipments in physical topology release their complex decision heuristics to the Controller, so they become simple, dummy equipments. However; on one hand, SDcN provides scalability and flexibility on network management with dummy OpenFlow (OF) switches and its nature of centralized authority; on the other hand, these properties cause spatial and temporal complexity in the Data Plane that should not be regarded according to Quality of Service (QoS) of a flow. Spatial complexity, described in this thesis as *Memory Usage* and/or *Hardware Cost* in an OF switch, should be minimized by removing redundancy in OF switch flow table pipeline. Temporal complexity, defined in this thesis as *Flow Forwarding Delay*, should also be reduced by lowering number of comparison in flow table pipeline to enhance QoS of a flow.

There are many studies in the literature that try to minimize spatial-temporal complexity in OF switch architecture and also to enhance QoS of a flow. However; most of these works violate dummy characteristic of OF switches because of the fact that they implement some wiser decision mechanism such as hashing, caching and search heuristics within the switch architecture.

Any mechanism offered to lower spatial-temporal complexity of OF switch architecture should not violate dummy characteristic of these devices. Otherwise, management of physical devices by a central controller becomes slower, more difficult and complicated. Therefore, we propose a novel OF switch architecture to lower spatial-temporal complexity by using Multi-stage Switch model in flow table pipeline. The proposed model is able to forward a flow by less header matching with its architectural model. The proposed model also protects dummy characteristic with a little alteration in OF protocol header fields. In order to define the next table in the pipeline of proposed Multi-stage model, an additional field called next table identifier is integrated with existing OF protocol header entries. Flow forwarding in each pipeline of Multi-stage model is performed according to this next table identifier. Moreover, the advantages of NoC paradigm are also applied to OF forwarding mechanism with such a Multi-stage switch model by assigning a microcontroller to each flow table in each of pipeline stages. Consequently, thanks to multiple pipeline

stages that can serve flows concurrently, the number of flows served per unit time is increased and the QoS of each flow is enhanced.

For the performance investigation of proposed model we consider Queuing Theory in the light of a spatial complexity parameter (C) and a temporal complexity parameter (T) defined in this work. Total memory usage in a flow table pipeline is measured with spatial complexity parameter and flow forwarding delay is meausered in terms of temporal complexity parameter. These parameters are considered during the performance comparison between conventional OF switch architecture and proposed MsOF switch architecture with $_{CON}$ and $_{MsOF}$ indexes respectively. According to performance evaluation results, $MsOF$ switch architecture is less spatial complex than conventional one as the number of input ports in an OF switch (N) increases that corresponds denser topologies. MsOF gains much more memory space by deploying more tables with lower memory sizes, total of $(n \cdot k)$. Moreover, $MsOF$ switch model has less temporal complexity compared to conventional OF switch model, especially in urban areas. With such a Multi-stage deployment for flow tables, proposed MsOF architecture provides approximately 7 times less $flow\ forwarding\ delay$, $T_{MsOF} < 7 \times T_{CON}$, compared with a conventional OF Switch by virtue of processing flows at different stages of pipeline simultaneously.

YAZILIM TABANLI HÜCRESEL AĞLAR (YThA) İÇİN UZAYSAL-ZAMANSAL VERİ KATMANI TASARIMI

ÖZET

Son yıllarda hızla artan IP trafik yoğunluğu ağ yönetimini zorlaştırmaktadır. CISCO'nun Görsel Haberleşme Ağı Göstergesi'ne (Visual Networking Index) göre, dünya çapında toplam IP trafiği geçtiğimiz beş yılda beş kattan fazla artış göstermiştir, ve önümüzdeki beş yılda bu değerin üç katına çıkacağı öngörülmektedir. Günümüzde var olan trafik kontrol mekanizmaları, aşırı artan trafik yoğunluğu karşısında esnek olmadığı için yeterli kaynak verimliliği sağlayamaz ve trafik yoğunluğunun neden olduğu ciddi servis kalitesi düşüklüklüklerini engellemede yetersiz kalır. Sonuç olarak, geleneksel ağ kontrol mekanizmaları, hızla gelişen mobil teknolojiler ve buna bağlı olarak artan veri trafiği karşısında kullanıcı gereksinimlerini karşılayamaz hale gelmiştir.

IP trafik yoğunluğunun günümüzde gelmiş olduğu bu noktada, trafik artışına karşı esnek olabilen ve fiziksel topolojinin karmaşıklığını ortadan kaldırarak ağı kolayca yönetebilen dinamik bir mekanizmaya ihtiyaç duyulmaktadır. Bu problemi çözebilmek için fiziksel ağ topolojisinin karmaşık yapısını azaltan ve merkezi bir yaklaşım ile tüm topoloji üzerinde hakimiyet sağlayarak ağı tek bir noktadan dinamik bir şekilde yönetebilen yeni nesil YThA teknolojisi geliştirilmiştir. Bu teknoloji ile anahtarların mantıksal kısmı kontrol katmanına taşınarak, bu cihazlar veri katmanında akılsız hale getirilip yönetimi kolaylaştırılmıştır. Veri katmanında yer alan fiziksel cihazlar ile kontrol katmanında yer alan ve bu cihazların yönetiminden sorumlu olan Kontrolör, YThA teknolojisi için tanımlanan OpenFlow protokolünü kullanarak haberleşmektedir. Böylece, YThA ve bu yapının kullandığı OpenFlow protokolü sayesinde, her fiziksel anahtarın birer birer konfigüre ve kontrol edilmesi yerine, fiziksel cihazların ağ kontrol birimi olan Kontrolör tarafından yönetimi kolayca sağlanmıştır. Ancak, YThA mekanizmasının ağ yönetiminde sağladığı esneklik ve fonksiyonelliğin yanında, OpenFlow anahtar yapısından kaynaklı olarak akış servis kalitesini kötü yönde etkileyen ve göz ardı edilmemesi gereken ek bir yönlendirme gecikmesi oluşmaktadır. Bu gecikmenin temel nedeni, bir akış yönlendirilirken OpenFlow anahtar yapısındaki akış tablosu iş hattında, birçok bitlik alanın birden fazla tabloda karşılaştırma ve eşleştirme gerektirmesidir. Geleneksel OF anahtarına giriş portları aracılığıyla iletilen bir akış, akış tablosu iş hattında yer alan tablolarda ilgili eşleşmelerden geçtikten ve başlık alanlarındaki bilgiler düzenlendikten sonra çıkış portları üzerinden tekrar ağa gönderilir. Akışın anahtar içerisinde geçirdiği süreçte, başlık alanları üzerinde herhangi bir değişiklik gerektirmeyen bazı eşleştirmeler de söz konusu olabilmektedir. Ayrıca, ağdaki anahtara iletilen akışların iş hattındaki ilerleyişi seri olarak devam etmektedir. Diğer bir deyişle, akış tablosu iş hattının tasarımından dolayı, iş hattında farklı akışlara paralel olarak hizmet verilememektedir. Geneksel OF anahtarında yer alan tüm bu mimari kısıtlar ve akış tablosu iş hattındaki tasarımsal eksiklikler, akışın yönlendirilmesi sırasında ilave gecikmeye neden olduğu gibi bellek kullanımı ve donanım maliyeti açısından da fazladan masrafa neden olmaktadır.

YThA teknolojisinde, anahtar mimarisinden kaynaklanan iki önemli dezavantaj belleksel karmaşıklık ve zamansal karmaşıklık olarak sıralanabilir. Bu tez çalışmasında, belleksel karmaşıklık mimaride yer alan donanım maliyeti ya da bellek maliyeti olarak tanımlanırken, zamansal karmaşıklık ise akış yönlendirmesi sırasında ortaya çıkan gecikmeyi işaret etmektedir. YThA teknolojisinin veri katmanında, anahtar yapısından kaynaklı belleksel ve zamansal karmaşıklığı azaltmak üzere, OpenFlow akış yönlendirme mekanizması ile ilgili literatürde birçok çalışma yer almaktadır. Bunların başlıcaları anahtar üzerindeki arama için Hash yönteminden yararlanma, arama verilerini önbelleğe alma, akış eşleşmesi aranırken mükemmel hash yöntemi kullanma olarak sıralanabilir. Ancak, bu ve benzeri arama yöntemleri kullanan ve bir karar mekanizmasına dayalı olarak ön bellekleme yapan çalışmalar, OpenFlow anahtarlarının çalışma yapısındaki akılsızlık özelliğini bozmaktadır. Halbuki akılsız anahtarların Kontrolör tarafından yönetimi daha kolay olduğu gibi, bu cihazların akış yönlendirmesi daha hızlıdır. Belirtilen problemlerin çözümü için sunulacak yöntem, hem OpenFlow anahtarının akılsızlığını bozmayan hem de akış yönlendirilmesinde daha az eşleştirme yaparak akışın servis kalitesini iyi yönde etkileyen ve gecikmeyi azaltan yapıda olmalıdır.

Ağda performans iyileştirmesi sağlarken OpenFlow anahtarının akılsızlık özelliğini bozmadan çalışababilecek bir yapı tasarlamak adına, bu tez çalışmasında, iş hattında yer alan tablolar çok katmanlı bir iş hattı mimarisi oluşturacak şekilde konumlandırılmıştır. Çok katmanlı anahtar mimarisi kullanarak, enine bağlantılı (crossbar) geleneksel mimariye göre, belleksel karmaşıklık azaltılmış ve anahtar yapısının donanımsal maliyeti düşürülmüştür. Önerilen mimaride iş hattında kullanılan çok katmanlı yapı 3 katmandan oluşmaktadır. Önerilen yapıda geleneksel OpenFlow anahtarının akış tablo dizilimindeki iş hattı yapısı değiştirilerek bir tablodan gidilebilecek tablo sayısı ikiye çıkartılmıştır. Anahtar mimarisinde yapılan bu değişikleri OpenFlow protokolü ile uyumlu hale getirmek adına, akışların başlık yapısında gerçekleştirilen ufak bir değişiklik ile iş hattındaki tablolar arası yönlendirmeyi sağlayan bir bitlik bir sonraki tablo belirteci eklenmiştir. Böylece, iş hattında gidilecek bir sonraki tablo, protokol ile belirlenirken, OpenFlow anahtarının akılsızlık özelliği korunmuştur. Ayrıca, çok katmanlı OpenFlow iş hattındaki tablolara birer mikroişlemci atanarak Yonga Üstü İletişim Ağı (Network On Chip-NOC) mekanizmasının sağladığı avantajlar OpenFlow anahtarına uygulanmıştır. NOC'ta, kullanılabilen birçok ortak yol (multi-bus) ve bu yollar üzerinde rezerve edilebilecek birçok mikroişlemci bulunduğu için anahtar içerisinde birim zamanda birden fazla akışa paralel olarak hizmet verilebilir. Geleneksel iş hattı yapısında bulunmayan bu özellik ile paralel akış işleme süreci elde edilmiştir. Bu sayede, akışlar arasındaki kaynak yarışını en aza indiren NOC tasarımı, akışın yönlendirme gecikmesini azaltarak servis kalitesinin arttırımında büyük ölçüde katkı sağlar.

Sonuç olarak bu tez çalışmasında, çok katmanlı mimari modeli için üç katmandan oluşan ağ mimarisi kullanılarak, anahtar içerisindeki akış tablosu iş hattı yapısı ve bu iş hattındaki tablo bağlantıları yeniden düzenlenmiştir. Ayrıca, anahtar mimarisi için Yonga Üstü İletişim Ağı yapısına benzer bir mimari oluşturmak ve bu mimarinin avantajlarından yararlanmak adına, her bir akış tablosuna, geleneksel yapıdaki tek bir mikroişlemci yerine, birim zamanda sadece bir komut gerçekleştirebilecek kapasitede birer mikroişlemci yerleştirilmiştir. Böylece, OpenFlow anahtarındaki bir iş hattı yapısı paralel çalışabilen birçok iş hattı yapısına dönüştürülmüştür. Ayrıca, anahtarların akılsızlık özelliğinin korunması adına, akışın çok katmanlı

ağ mimarisindeki hareketi, ilgili akış tablo satırına bir sonraki tabloyu belirten bir bitlik belirteç eklenmesi ile gerçeklenmiştir. Geleneksel anahtar yapısının ve önerilen anahtar yapısının performans incelemesi ve karşılaştırması için anahtar içerisindeki iş hattı mimarisini modellerken kuyruklama teoreminde yararlanmıştır. Her iki mimaride yer alan iş hattı bölümü kuyruk ve sunucudan oluşan bir yapı ile modellenmiştir. Kullanılan modelde, her bir tablo, akışa hizmet veren ve eşleştirme işlemini gerçekleştiren bir sunucu olarak düşünülmüştür. Geleneksel yapıda ve önerilen yapıda kullanılan iş hattı tasarımına göre modelin uygun bölümlerine kuyruk eklenmiştir ve belirli bir varış oranında anahtara gelen akışlar bu kuyruklarda bekletilmiştir. Kuyruklama teoremi kullanılarak oluşturulan modelde, belleksel karmaşıklık ve zamansal karmaşıklık ölçütü olarak C (C_{MsOF}, C_{CON}) ve T (T_{MsOF}, T_{CON}) adında iki yeni parametre tanımlanmıştır ve geleneksel mimari ile önerilen mimari arasındaki performans incelemesi bu iki parametere üzerinden yürütülmüştür. Belleksel karmaşıklık parametresi ile anahtar mimari yapısının donanım maliyeti formülize edilirken; zamansal karmaşıklık parametresi ile akışın yönlendirme gecikmesi matematiksel olarak ifade edilmiştir. Elde edilen performans sonuçlarına göre, çok katmanlı (3 katmanlı) iş hattı yapısında kullanılan toplam bellek miktarı ve donanımsal bağlantı maliyeti geleneksel anahtar mimarisine göre azalmıştır. Diğer bir deyişle, geleneksel mimaride yer alan belleksel karmaşıklık azaltılmıştır. Ayrıca, önerilen switch modelinin geneleneksel switch modelinden, özellikle kalabalık ağlarda, yedi kata kadar daha az yönlendirme gecikmesi ile akış yönlendirebildiği ve daha iyi hizmet kalitesi (QoS) sağladığı görülmüştür. Böylece, geleneksel mimarinin akış kalitesini köte yönde etkileyen zamansal karmaşıklığın önerilen mimaride azaltıldığı gözlemlenmiştir. Bu çalışmadan elde edilen sonuçlara göre, tabloların dizilişi çok katmanlı ağ mimarisine göre tasarlanarak, akış yönlendirmesi daha az eşleştirme ile gerçekleştirilmiştir. Böylece anahtar sisteminin belleksel ve zamansal karmaşıklığı azaltılarak performans iyileştirmesi sağlanmıştır.

1. INTRODUCTION

In this chapter, the purpose of work done in this thesis, literature review for similar studies and also for the methodology used, and hypothesis given in the thesis are introduced, respectively.

1.1 Purpose of Thesis

Recently, cellular mobile data traffic has tremendously increased [4]. This rapid increase, arising from various mobile technologies, gives rise to stressed conventional cellular networks. Software Defined Cellular Networking (SDcN) is one of the novel approaches offered to alleviate these crowded cellular networks on physical topology. Due to its central management fashion of dummy devices located in Data Plane, provided scalability and flexibility properties on the topology makes it easier to orchestrate cellular networks [5]. However, SDcN has some significant challenges that cause spatial complexity indicating the memory usage and temporal complexity illustrating flow forwarding delay because of OF switch architecture in Data Plane.

The purpose of this thesis is to design a novel OpenFlow switch model for SDcN Data Plane that lowers spatial-temporal complexity of the current one, namely reduces the memory usage and flow forwarding delay. The proposed switch model should communicate with Control Plane using an extended OpenFlow protocol. At the same time, working mechanism of OpenFlow protocol and dummy characteristic of physical devices in Data Plane should be maintained.

1.2 Literature Review

Literature review on hardware cost and flow forwarding delay of OF switch architeure, namely the spatial-temporal cost, is represented in this section. Moreover, NoC paradigm and multi-stage switch architecture deployed in this work are also reviewed in the literature.

1.2.1 Spatial-temporal compexity of OpenFlow switch model

Spatial complexity in SDcN Data Plane is defined as memory usage (in bits), i.e. cost of hardware. An OpenFlow (OF) switch needs to be dummy according to SDcN fashion and is orchestrated by SDcN Controller to provide scalability and flexibility on the topology. However, the working mechanism of conventional OF switches in Data Plane brings about higher spatial complexity especially in urban areas [6]. As a dummy device, an OF switch looks up an incoming flow in its flow table pipeline in order to set actions of this flow. In this procedure, fields of the incoming flow is compared with each row of each flow table in the pipeline, and eventually, an exact match must be provided in order to route and forward the incoming flow [7] [8]. In other words, each flow table in the pipeline perform a searching between incoming flow fields and its entries. When a hit occurs in the search, the corresponding action of the flow is set by a predifined action in the table. However, there may some redundant searching that requires no action. Even for this situation, each table must provide a predefined *goto next* entry in itself that means the result of the matching will not change the action set of incoming flow. Therefore, this kind of actions that requires nothing to set in the incoming flow actions waste some memory. This nature of conventional OF switches cause a spatial complexity, i.e. redundant hardware cost.

Temporal complexity is defined as flow forwarding delay (in seconds), i.e. Quality of Service (QoS) of flow. In urban areas, it is high for SDcN paradigm because of two main reasons. One of them is the bottleneck arising from central management strategy and this bottleneck brings extreme delay on QoS of flow, especially in urban areas [9]. Furthermore, the temporal complexity is also caused by OF switch working mechanism. This complexity is also high because of the fact that each flow has to perform looking up with each flow table in the pipeline. Moreover, each flow actions are executed on only one processor at the end of the pipeline. Therefore, the required time to forward a flow has extra redundant delays on both looking up on many flow tables and waiting on one processor's queue. The architecture and working mechanism of SDcN and OF switches becomes more complex in terms of temporal complexity as the number of users in the topology rises.

There exists many studies in the recent literature to overcome spatial-temporal complexity of Data Plane. The studies on spatial complexity try to decrease memory

usage in the system in order to lower hardware cost. For example in [10], the authors study on a novel switch design with different memory management policies. They examine average message latency according to different switch design parameters such as memory page size and cache block size. In [11], a new model for Deep Packet Inspection is proposed to minimize on-chip memory space and increase performance of network processors. In paper [12], spatial and temporal aspects on memory parallel access efficiency are studied and costs of storage space is tried to minimize with multi-core architecture. On the other hand, most of the researches in the literature try to accelerate OF switches in Data Plane by providing less temporal complex architectural models. In paper [13], a novel switch cache architecture for cache-coherent NUMA multiprocessors is proposed by using cache memories in the crossbar switch. Therefore, the authors assert that memory access latency and the bottleneck challenges can be removed. In paper [14], proposed cache embedded switch architecture results in a reduction in the inter-cache transfer time and the total execution time in on-chip multiprocessor platform. [15] offers Hashing strategy, a commonly used approach, for OF switches in order to quicken search on flow tables. In [8], frequently used entries of flow tables moved into a rapid cache memory. Moreover, [8] investigates performance measurements between proposed caching strategy, hashing strategy and linear search on flow tables. With a flexible flow forwarding approach, [16] offers an innovative structural design for OF switches and use Perfect Hashing in order to provide an amendment on temporal performance. Furthermore, [17] considers traffic types of each flow in the queue of an OF switches and assign channels correspondingly. They also propose a novel switch architecture named Combined Input Crosspoint Queued (CICQ) that provides a better flow performance in the topology. In that study, the spatial complexity is also decreased, but not analytically examined in detail.

In this thesis, existing studies in the literature are considered, but an extensive performance comparision between them cannot be achieved because of the absence of simulation data similar to the one in this work. Only the switch architecture specified in OpenFlow Protocol Specification is considered and its performance results is compared with the proposed model in terms of spatial complexity parameter and temporal complexity parameter defined in this thesis.

1.2.2 NoC model in literature

In this work, NoC paradigm is inspired for flow table pipeline architecture because of its advantages over traditional single bus-based architectures. It is considered to assign a micro-controller to each OpenFlow table in the pipeline for processing of a flow. Whereby, a kind of NoC mechanism is adapted to flow table pipeline of proposed switch architecture in order to take the advantages of this technology in hardware level indicated as [18]. In such an architecture, there are multiple bus opportunities for data to be sent and that characteristic provides performance enhancement in terms of throughput and QoS stated in [19] and [20]. Such a switch architecture is also able to reduce race condition between flows for physical resources, especially for processor unit. Moreover, parallel processing of flows could be utilized with the help of such an architecture. Thus, QoS achieved in the topology gets better as in the relation between NoC technology and system throughput stated in [7]. A multi-pipeline architecture is deployed in this work that consists of multiple pipelines running simultaneously by adapting this technology to flow table pipeline.

1.2.3 Multi-stage switch model in literature

Multi-stage switch architecture is examined in order to reduce spatial-temporal complexity of current OF switch architecture. In conventional crossbar architecture, each line is connected to another one at their cross point. Such an architecture ends up with a state explotion for large number of lines in the switch. However; multi-stage architecture offers to gruop lines into k-stages where each stage has a crosspoint architecture within and different stages are connected to each other with k connection points. [21] states the advantages of multi-stage switch architecture over traditional crossbar switch architecture. According to the author, a lower spatial complexity is obtained by implementing a k-stage architecture compared to applying crossbar interconnections.

Considering spatial and temporal superiority of multi-stage architecture, a special kind of this paradigm including 3 stages is surveyed for proposed switch design. In this model, there are 3 stages named as Stage-1, Stage-2 and Stage-3 in the architecture. Each table has 2 inter-connections to the corresponding table of the following stage. With usage of this architecture in the proposed switch design, spatial complexity is

reduced with lower number of connections and a temporal improving is obtained for flow forwarding with simultaneous processing in Stage-1 and Stage-3.

1.3 Hypothesis

Caching strategies, hashing on flow tables and search algorithms applied in the aforementioned studies need some algorithmic decision mechanisms. For this reason, these studies in the literature violate dummy characteristic of OF switches in Data Plane. However; SDcN fashion requires dummy physical devices in order to manage them all in one central controller and provide faster forwarding capability even in urban areas. Therefore, any solution to be proposed for mentioned problems should not violate the dummy characteristic of OF switches and provide an inventive switch architecture to reduce spatial-temporal complexity of SDcN fashion.

In order to solve aforementioned challenges of SDcN technology, we propose a Multi-stage OF (MsOF) switch model that does not violate the dummy characteristic of switches in Data Plane and forwards a flow with less matching in flow table pipeline. Due to multi-stage architecture, the cost of memory usage can be less than conventional OF switch pipeline architecture that corresponds a lower spatial complexity. Moreover, thanks to this novel architecture, there could be multi flow table pipeline which can forward flows in parallel. Therefore, it enhances QoS of flow by decreasing flow forwarding delay that is defined as temporal complexity in the architecture.

The proposed construction details of the novel MsOF switch model are given as follows:

- The intelligence mechanism of OF switches in Data Plane is moved to virtual representaters of them in Control Plane.
- Flow table ordering and locating in the pipeline are organized according to multi-stage network model to forward a flow in at most three hops. Hence, a 3-stages architectural model is deployed for flow table pipeline in OF switch.
- A Batcher is employed to appropriately arrange incoming flows going through 3-stages flow table pipeline.

- Forwarding of a flow between tables in the pipeline comes true according to *next table identifier* that is added to existing flow table fields. Thus, movement of a flow in the pipeline is determined by a field in protocol headers and dummy property of OF switch is maintained.

- Instead of using one processor at the end of pipeline, a microprocessor is deployed for each flow table in multi-stage switch pipeline. These microprocessors have a service rate of performing an action per unit time.

- The pipeline architecture in conventional OF switch is altered by multi-pipelines that can be work in parallel and serve flows at the same time.

2. CONVENTIONAL NETWORK ARCHITECTURE

Sample network architecture for SDcN technology is given in Figure 2.1. There are two main components in SDcN fashion named as Control Plane and Data Plane. Communication and control signals between these planes occurs according to OpenFlow Protocol which is standardized in SDcN paradigm. Details of these planes are examined with following sections.

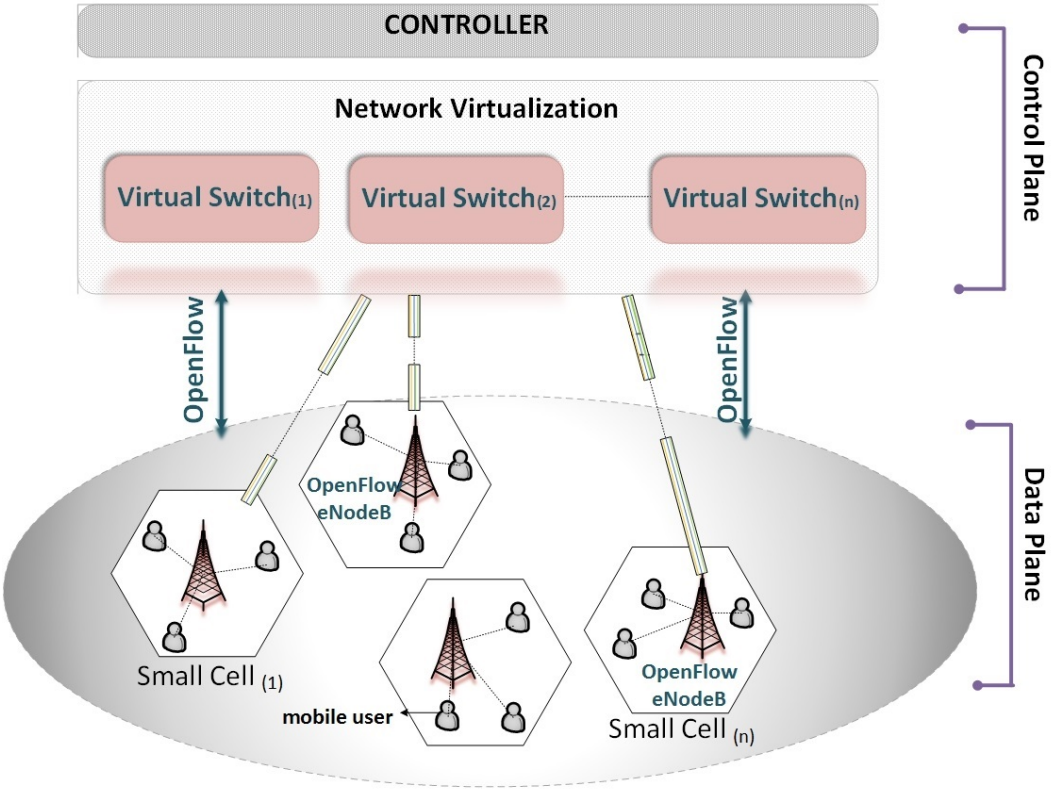

Figure 2.1: Sample SDcN Topology.

2.1 Control Plane

Control Plane of the topology consists of a Virtual Layer containing the virtual reflection of Data Plane and a Controller. There is a corresponding virtual switch in Control Plane for each of OF switches in Data Plane. These virtual switches in Virtual Plane are created while the topology is established for the first time and the

Virtual Plane is configured each time an update in Data Plane occurs. The controller is responsible for orchestrating of virtual switches. All managerial actions are appeared in control plane between controller and virtual switches. These decisions are conveyed to each OF switch in Data Plane by corresponding virtual switch in Control Plane using OpenFlow Protocol. With such a cellular network deployment, control and forwarding functions are decoupled using two separate planes by virtue of SDcN fashion.

2.2 Data Plane

There are pyhsical OF switches located in Data Plane of an SDcN topology. These switches are lack of the ability of decision making. These devices are dummy with only forwarding capabilities. Flow tables are located in an OF switch that process forwarding operation of an incoming flow. OpenFlow Protocol sets the standarts for forwarding operations. Inner architecture and components of an OF switch in Data Plane are detailly explained in Chapter 4.

2.3 OpenFlow Protocol

OF protocol is the standard communication interface defined between control and forwarding layers of SDcN architecture. The Controller communicates to OF switches using a secure channel (see Figure 2.2). OF switches contain flow tables which are used for forwarding and packet header look-ups. OF protocol makes switches able to update their flow table entries in order to take corresponding actions according to incoming flow and matching entries. The control channel is responsible for updating these flow tables. In addition, dummy switches forward a flow according to their flow tables, those are installed by the Controller [1].

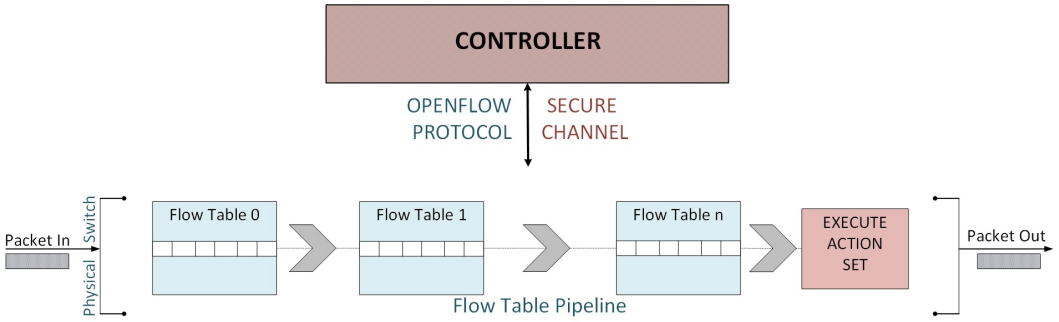

Figure 2.2: Idealized OF Switch [1].

The flow table contains some flow entries. Those are counters and a set of actions to be applied for matching packets. These packets are processed by the OF switch using the flow tables, as depicted in Fig.2.3. If a matching entry is found, the corresponding action on the packet will be set and performed at the end of the pipeline. If it does not match, the packet will be forwarded to the controller to determine how to handle it.

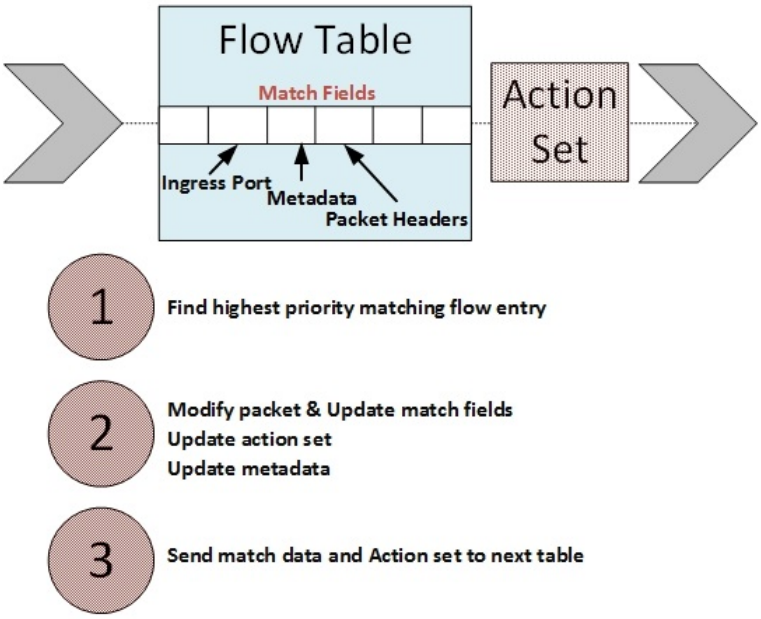

Figure 2.3: Steps of a Packet Flow [2]

There can be many flow tables in an OF switch. Therefore, packets are matched against multiple tables in the pipeline. Figure 2.2 illustrates a pipeline processing for a packet residing in the OF switch. Tables in the pipeline can update packet header fields and add the corresponding set of actions that will be performed before the packet is sent to output ports. This set can be empty. A flow table consists of three main fields as shown in Figure 2.4. Match fields contain packet headers and ingress ports. Counters are used to update matching packets, and instructions allow the system to modify the action set that will be applied to an incoming packet.

In SDcN, the default flow management is handled as follows [22]: All incoming flows received by the OF switches. The switches check destination address of each flows and if the destination address could not be matched with any flow table entries in the switch, it is defined as newcomer flow and forwarded to the to Controller [23]. Here, the controller is responsible for updating the flow table entry and assign a forwarding rule for this newcomer flow. Consequently, with the result of virtualization

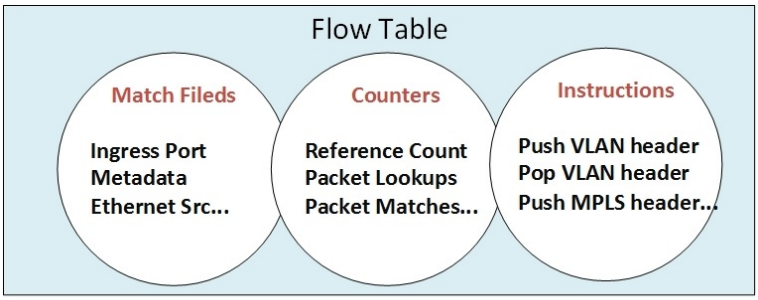

Figure 2.4: Main Fields of a Flow Entry in a Flow Table [3]

algorithms, the Controller decides about the new forwarding entry and updates the flow tables in switches by considering fair flow distribution. These forwarding messages are disseminated using the OpenFlow Protocol to the entire network. However, this centralized decision characteristic causes both a flow load bottleneck and latency on overall flow traffic, bringing a communication overhead on quality of flows [24], [25]. In order to remove this negative effect, there exist some solutions. For example, in order to overcome the communication overhead between the data and control plane, [26] proposes three extensions to OpenFlow Protocol headers and flow table fields. They claim that these approaches will accelerate the production in networks. [27] enhances the flexibility and configuration of system to save more energy by deploying the OpenFlow Protocol extension to their cloud architecture. In this thesis, a similar extension is provided to OpenFlow Protocol headers for proposed flow table pipeline model in MsOF switch architecture.

3. PROPOSED SYSTEM MODEL

Wireless network topology architecture considered in this work is given with Figure 3.1. Control Plane, Data Plane and OpenFlow protocol of this SDcN topology is explained in following sections.

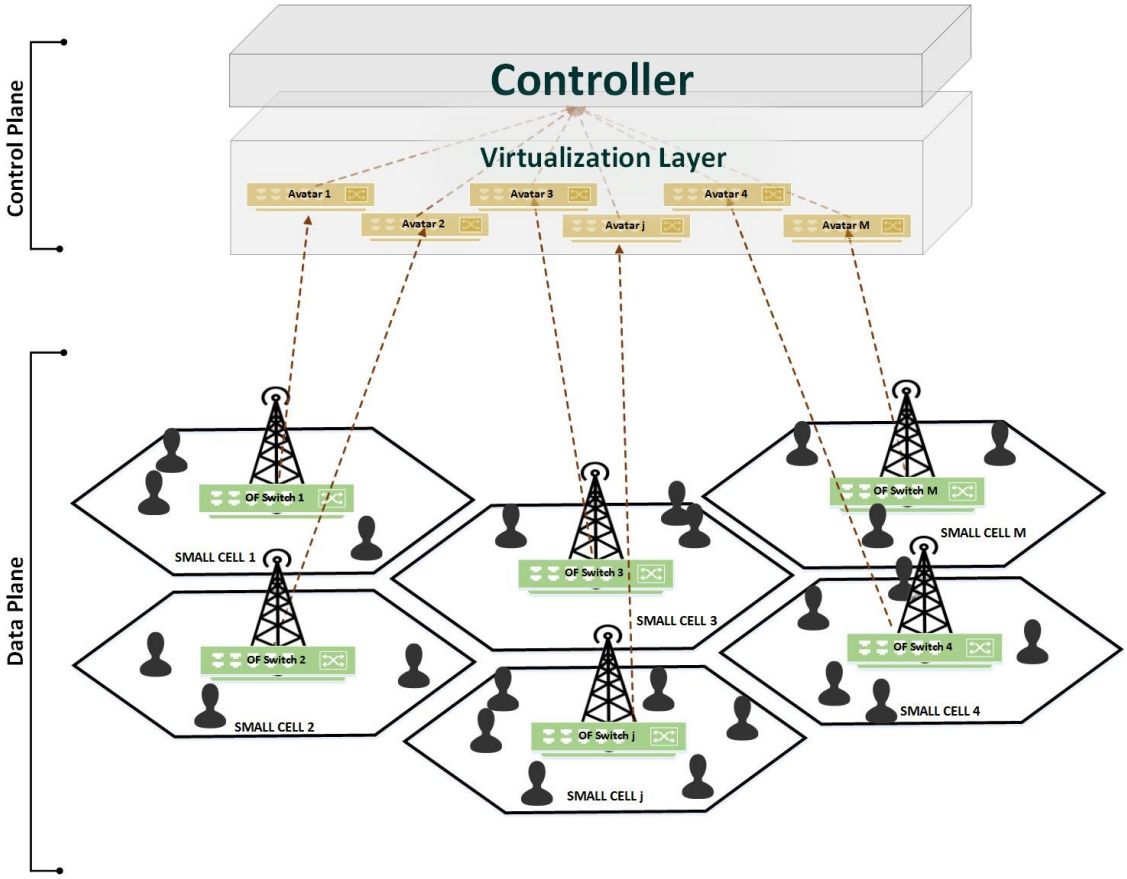

Figure 3.1: MsOF Network Architecture.

3.1 Control Plane

As usual, Control Plane of the SDcN topology studied in this work includes two main components named as Controller and Virtualization Layer. Controller as the mastermind of the system is responsible for monitoring virtual switches, named as Avatars in this thesis, in Virtualization Layer. Virtualization Layer includes virtual representatives of pyhsical devices in Data Plane and responsible for transmission of

appropriate signals coordinated by the Controller. This plane in the architecture is not specialized with an extra work and considered as described in Section 2.1. Main contribution given in this work is focused on Data Plane.

3.2 Data Plane

For Data Plane of studied SDcN topology in this work, a novel MsOF switch is offered. A detailed inner architecture of this switch including input/output ports and software/hardware components is given with Figure 3.2. Components of software and hardware parts of the switch are explained with following subsections.

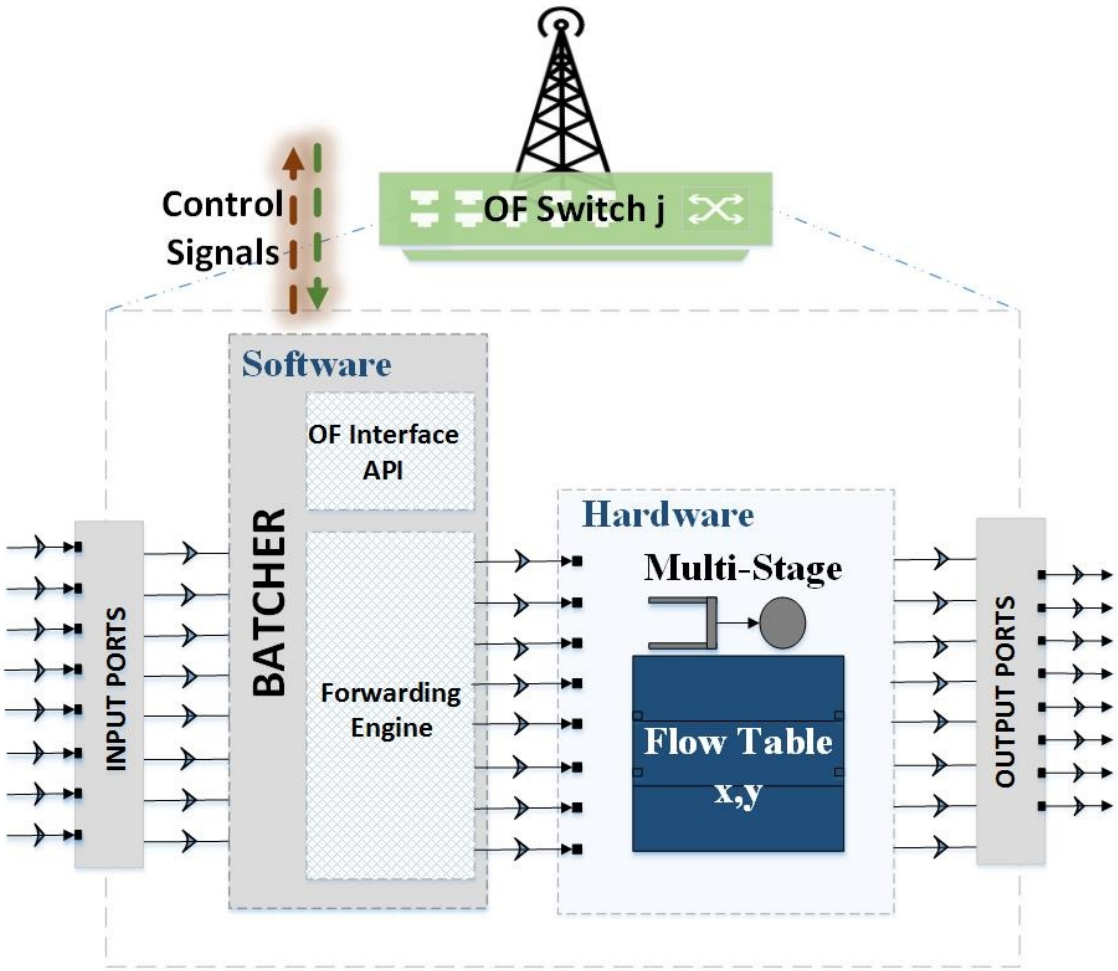

Figure 3.2: Proposed MsOF Switch Architecture.

3.2.1 Software components of multi-stage OpenFlow switch

A batcher exists in the model as a software component. This part is responsible for arrengement of incoming flows with two sub-parts named OF Interface API and

Forwarding Engine. Incoming flows are transmitted to flow table pipeline by this part in order to be processed.

3.2.1.1 Batcher

OpenFlow interface API:

This unit communicates with the Avatar of OF switch in the Control Plane. By configuring OpenFlow Protocol using OF Interface API, extracted data can be taken and sent to Forwarding Engine sub-module of Batcher. The opposite way communication is also supported when any fault occurs or there is a newcomer flow to the switch.

Forwarding engine:

It is responsible for implementing configurations on hardware (multi-stage) side of a switch and also responsible for switching between OF switch modes that are sent from the corresponding Avatar. According to control commands taken from Avatar, the flows taken from input ports are suitably ordered and forwarded to multi-stage flow table pipeline by Forwarding Engine sub-module.

3.2.2 Hardware components of multi-stage OpenFlow switch

3.2.2.1 Input output ports

The flows reach to the batcher coming through N number of input ports with a total inter-arrival rate of $\sum \lambda_j(t)$ where j is the index of corresponding OF switch and $\lambda_j(t)$ is the inter-arrival rate of a single OF switch. After the flows are organized by *Physical Layer Processor* and then arrenged by *Data Link Layer Processor*, they enter input queues to wait for the completion of previous flow processing inside the OF switch.

After the forwarding procedure and required actions are applied to flows, they are sent to network via N number of output ports. In output ports, firstly, the flows are organized by *Data Link Layer Processor*. Subsequently, by *Physical Layer Processor*, the flows enter in output queues. After these procedures are completed, they are ready to transfer to network link.

3.2.2.2 Multi-stage switch model

The proposed OF switch flow table pipeline includes three stages as seen in fig. 4.2. In each stage, there are flow tables which are located and filled according to corresponding Avatar's control commands. Each flow table has a processor with a predefined service rate of μ_{MsOF}. These processors can execute one action per unit time.

According to OpenFlow Protocol Specification 1.4, each flow requires at least 2 actions to be forwarded [5]. Due to the fact that there can be another actions to be applied for a flow, the design of OF switch is reorganized considering 3 hops in 3 stages. If the number of actions for a flow exceeds 3, the remaining actions are executed in $Stage_3$ in the processors. Therefore, the queue length of processors at $Stage_3$ is assumed as infinite. There are $\frac{N}{n}$ flow table in $Stage_1$ and $Stage_3$. Each table size is $n.k$. However, $Stage_2$ differs from other stages. It has k flow tables with size of $(\frac{N}{n})^2$.

The flow, forwarded from input ports to multi-stage flow table pipeline by the Batcher, looks up all headers to consider an exact match in flow table entries. After, the corresponding action is read from the matched row, it is executed by processor and forwarded to next table considering the *next table identifier* as explained in Section 3.3. Therefore, a flow can be forwarded by searching at most three flow tables.

3.3 OpenFlow Protocol

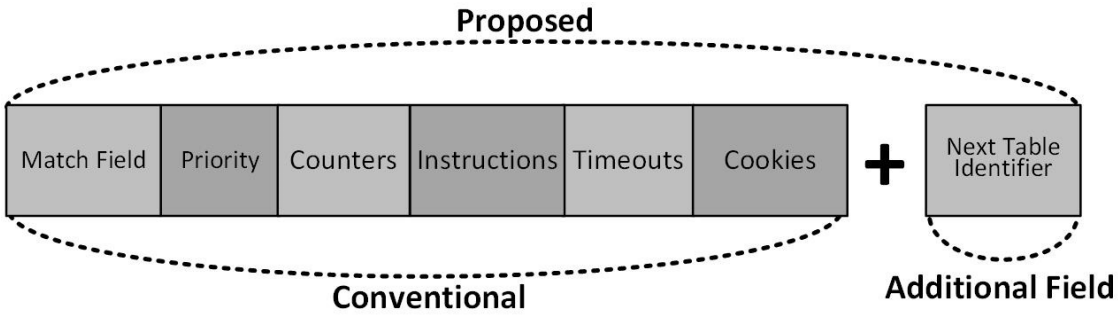

Figure 3.3: Fields of a Flow Table in Proposed Architecture.

The extended OF Protocol headers and a flow table row is reorganized as shown in Figure 3.3. The *next table identifier* field is added on conventional OF switch flow table fields. This field requires a length of $log_2(k)$ bits in $Stage_1$, $\log_2(\frac{N}{n})$ bits in

$Stage_2$ and is empty in $Stage_3$ due to being last stage. Therefore, the length of this field is defined as $\log_2(\frac{N}{n})$ because of being biggest value.

According to this field, a flow can be forwarded to the suitable table in next stage which is predetermined by the corresponding Avatar in Control Plane. Therefore, this intelligent forwarding strategy of proposed scheme ensures the dummy property of OF switch and keeps the delay for passing between flow tables at acceptable levels.

Employing multi-stage flow table pipeline architecture is more advantageous compared to the one with crossbar connections considering memory usage that corresponds to the spatial complexity of the system. If the crossbar approach is used on location design of flow table, there would be much more next table that must be identified. This will increase the size of *next table identifier* field in terms of bits. Due to prevent the memory demand of extra added fields, the multi-stage design is used in the proposed OF switch forwarding engine.

4. SPATIAL-TEMPORAL ANALYSIS

In this chapter, spatial complexity parameter and temporal complexity parameter are presented in order to be used in performance evaluation. In order to compare MsOF switch and conventional OF switch theoretically, queuing theory is studied. At this point, spatial complexity paremeter considering memory usage and temporal complexity parameter considering the flow forwarding delay of each switch architectures are obtained. Each of the systems is considered as multi $M/M/1$ model which are solved according to Jackson's theorem [28].

4.1 Conventional OpenFlow Switch

Queuing model of the conventional OF switch flow table pipeline can be shown in Figure 4.1. To forward a flow, OF switch pipeline must progress through the following steps:

- Each header field of a flow must exactly match with all fields in a row of a flow table. The number of bits that must be matched in a flow table, is defined as b_{CON} and the delay for exact matching in a row is defined as t_r.
- It is necessary for a flow to pass each flow table due to protect pipeline rule specified in the protocol.
- Each time that a flow is matched in a flow table, the action set which is initially empty, is filled one by one according to matched header action subfields. If the action is "*goto next*", the flow is sent to next table without any change on its header fields or action set.
- All actions in the action set of a flow are executed in the *Action Execution Server* located at the end of pipeline. The service rate of the server is defined as μ_{CON2} as seen in Equation 4.3.

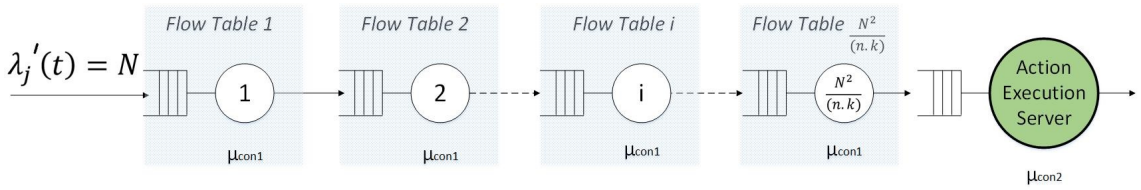

Figure 4.1: Conventional OF Switch Model.

Analysis of spatial complexity and temporal complexity of conventional OF switch table pipeline model will be demonstrated in following subsections.

4.1.1 Spatial complexity analysis

Due to each flow must pass through each flow table in the pipeline (according to dummy property of OF switch), conventional OF switch pipeline leads a repetition on the same matching fields in many flow tables. Therefore, the cost (memory usage) of conventional OF switch increases that corresponds a high spatial complexity.

Consider a flow table pipeline in a conventional OF switch with totally N^2 number of rows and also consider that each individual table in this pipeline has $n \cdot k$ number of rows. There will be $\frac{N^2}{n \cdot k}$ flow table in the switch where N is the number of input ports and $n \cdot k$ is the flow table size. In other words, it has $\frac{N^2}{n \cdot k}$ number of $M/M/1$ systems which are ordered in OF switch pipeline architecture. It is also assumed that the total number of bits in a row is shown with b_{CON}. At the end of these definitions, the total hardware cost of conventional OF switch that corresponds to the spatial complexity parameter can be calculated as:

$$C_{CON} = N^2 \cdot b_{CON} \qquad \forall N \in 1, 2, ... \tag{4.1}$$

4.1.2 Temporal complexity analysis

Again, due to each flow must pass through each flow table in the pipeline for presenting dummy condition, the number of hops per flow is exceeded in conventional OF switch pipeline in urban areas. Therefore, the flow forwarding delay becomes high and negatively affects the QoS of a flow.

As seen in Figure 4.1 and indicated in Section 4.1.1, there exists $\frac{N^2}{n \cdot k}$ number of $M/M/1$ systems and a server with service rate of μ_{CON2} at the end of the pipeline

in conventional OF switch. Looking up procedure of each flow table is modeled as a server. The server rate for a flow table is calculated as:

$$\mu_{CON1} = \frac{1}{n \cdot k \cdot t_r} \tag{4.2}$$

where $n \cdot k$ is the memory size of a flow table and t_r is the time required for exact matching in a row.

The server at the end of the pipeline that executes the actions of corresponding flow, completes its job in approximately t_{CON2} seconds. Therefore, the service rate of this server can be shown as:

$$\mu_{CON2} = \frac{1}{t_{CON2}} \tag{4.3}$$

On the other hand; according to Little's Law [28], the number of flows in the system is defined as:

$$N'_j(t) = \lambda'_j(t) \cdot T'_j(t) \tag{4.4}$$

where j shows the index of the OF switch, $N'_j(t)$ is the total number of flows, $\lambda'_j(t)$ is the total arrival rate of flows and $T'_j(t)$ is the flow forwarding delay in j^{th} OF switch at time t. In order to obtain the flow forwarding delay $T_{CON}(t)$ (which is represented $T'_j(t)$ in this subsection), firstly the total number of flows $N'_j(t)$ in the system should be found.

$N_j(t)$, the number of flows in one $M/M/1$ queuing system, is calculated as in following formula [28]:

$$N_j(t) = \frac{\rho_j(t)}{1-\rho_j(t)} \qquad \forall \rho_j(t) < 1 \tag{4.5}$$

where $\rho_j(t)$ is the utilization of one $M/M/1$ system in terms of $\lambda_j(t)$, μ_{CON1}, μ_{CON2} and μ_{MsOF}.

We investigate total number of flows in a OF switch considering the number of flows in a flow table and the number of flows in the Action Execution Server within following parts:

1. PART I: The total number of flows at flow tables of an $M/M/1$ system can be calculated as follows:

$$N_j^{1st}(t) = \frac{\lambda_j(t) \cdot n \cdot k \cdot t_r}{1 - \lambda_j(t) \cdot n \cdot k \cdot t_r} \tag{4.6}$$

where the $\lambda_j(t) = \lambda_j^{'}(t)\ flow/sec.$

2. PART II: The number of flows at the action execution server of an $M/M/1$ system can be calculated as follows:

$$N_j^{2nd}(t) = \frac{\lambda_j(t) \cdot t_{CON2}}{1 - \lambda_j(t) \cdot t_{CON2}} \tag{4.7}$$

where the $\lambda_j(t) = \lambda_j^{'}(t)\ flow/sec.$

By using the equations 4.6 and 4.7, the total number of flows in whole OF switch system is found as:

$$N_j'(t) = \frac{N^2}{n \cdot k} \cdot N_j^{1st}(t) + N_j^{2nd}(t) \tag{4.8}$$

$$N_j'(t) = \lambda_j'(t)[\frac{N^2.t_r}{1 - \lambda_j'(t).n.k.t_r} + \frac{t_{CON2}}{1 - \lambda_j'(t).t_{CON2}}] \tag{4.9}$$

As a result, according to Equation 4.4, the flow forwarding delay $T_{CON}(t)$ is obtain as follows:

$$T_{CON}(t) = T_j'(t) = \frac{N^2.t_r}{1 - \lambda_j'(t).n.k.t_r} + \frac{t_{CON2}}{1 - \lambda_j'(t).t_{CON2}} \tag{4.10}$$

4.2 Multi-Stage OpenFlow Switch

Flow table pipeline architecture with queuing model of proposed MsOF switch design is shown in Figure 4.2. Analysis of spatial complexity and temporal complexity of this model will be illustrated in following subsections.

4.2.1 Spatial complexity analysis

As seen in Figure 4.2, there exists 3 stages in MsOF switch architecture. In the first stage, there are $\frac{N}{n}$ number of flow tables, each has $n \cdot k$ number of rows where N is the number of input ports and $n \cdot k$ is the flow table size. In the second stage, there are k number of flow tables, each has $(\frac{N}{n})^2$ number of rows. The last stage is the same as the first stage in terms of flow table memory size. Moreover, each row's length is b_{MsOF} bits. The difference between b_{MsOF} and b_{CON} equals to $log_2(N/n)$ bits because of the

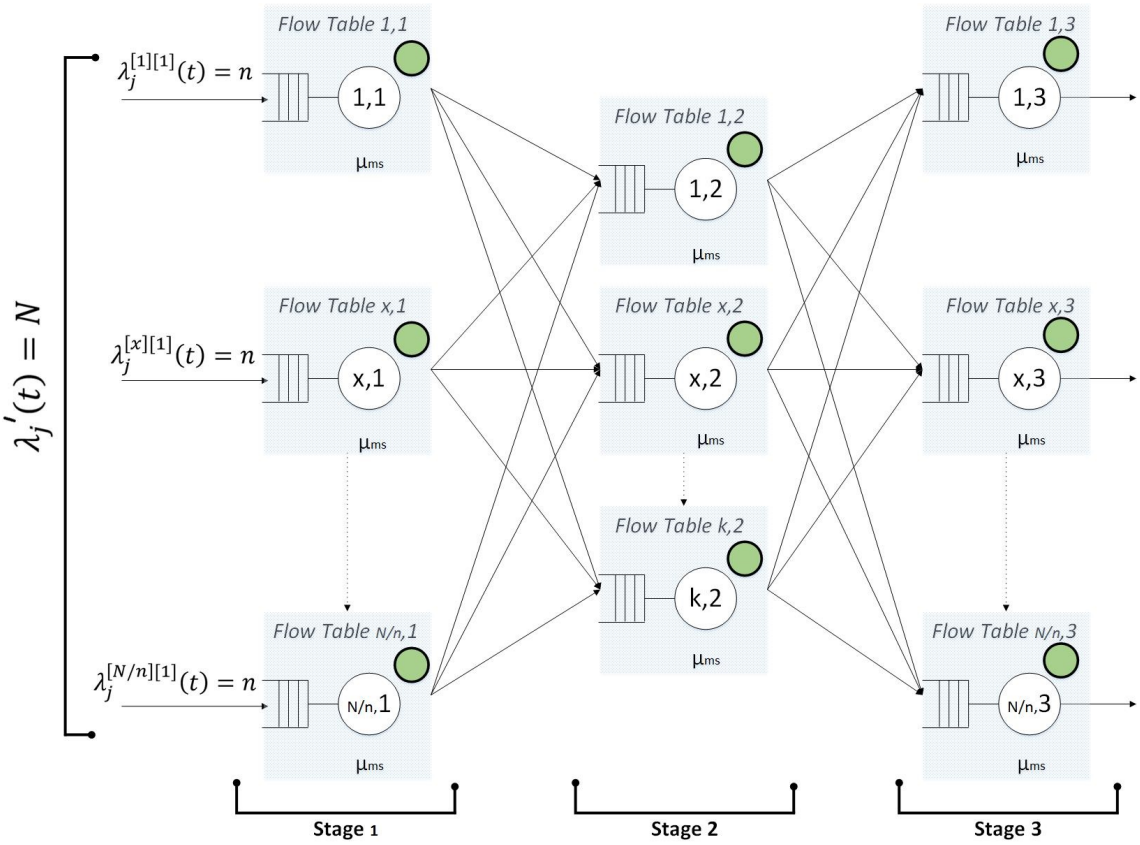

Figure 4.2: MsOF Switch Model.

added *next table identifier* field into flow table fields as explained in Section 3.3. At the light of these definitions, the total cost of proposed MsOF switch is calculated as in following equations:

$$C_{MsOF} = b_{MsOF} \cdot [2 \cdot \frac{N}{n} \cdot (n \cdot k) \cdot + k \cdot (\frac{N}{n})^2] \tag{4.11}$$

$$C_{MsOF} = b_{MsOF} \cdot N \cdot k \cdot (2 + \frac{N}{n^2}) \tag{4.12}$$

This equation is similar with the one in conventional OF swtich except including 3-stages. All of these stages are computed individually and combined at the end.

4.2.2 Temporal complexity analysis

To obtain $T_{MsOF}(t)$ which is represented here as $T_j'(t)$, the same procedure of $T_{CON}(t)$ extraction will be applied. However, each stage in MsOF Switch model is examined separately as different $M/M/1$ systems. The incoming flow rates for each $M/M/1$ system are represented with $\lambda_j^{[x][y]}(t)$ where x indicates the index of port numbers and y shows the index of stages. The number of flows in an $M/M/1$ system is shown with

$N_j^{[x][y]}(t)$ where x and y are aforementioned indexes. The probabilities of passing from an $M/M/1$ system of $Stage_1$, to another $M/M/1$ system of $Stage_2$ and similarly, from $Stage_2$ to $Stage_3$ are predefined as $p_{1to2} = \frac{1}{k}$ and $p_{2to3} = \frac{n}{N}$, respectively.

Moreover, each $M/M/1$ system service rate is shown as μ_{MsOF} and calculated as:

$$\mu_{MsOF} = \begin{cases} \frac{1}{t_{MsOF}+(n\cdot k)\cdot t_r}, & \text{for } Stage_1 \text{ and } Stage_3 \\ \frac{1}{t_{MsOF}+(\frac{N}{n})^2\cdot t_r}, & \text{for } Stage_2 \end{cases} \tag{4.13}$$

where t_{MsOF} is the delay of a processor in each $M/M/1$ system. The other parts are coming from exact matching delay in flow tables considering 3 different stages.

To obtain $T_j'(t)$, firstly $N_j'(t)$ should be found. It will be obtained from 3 different stages separately using the following equation:

$$N_j'(t) = \sum_{x=1}^{N/n} N_j^{[x][1]}(t) + \sum_{x=1}^{k} N_j^{[x][2]}(t) + \sum_{x=1}^{N/n} N_j^{[x][3]}(t) \tag{4.14}$$

For Stage$_1$, by using Equation 4.13:

$$N_j^{[x][1]}(t) = \frac{\lambda_j^{[x][1]}(t)\cdot(t_{MsOF}+(n.k).t_r)}{1-\lambda_j^{[x][1]}(t)\cdot(t_{MsOF}+(n.k).t_r)} \tag{4.15}$$

For Stage$_2$, the arrival rate per $M/M/1$ system is:

$$\lambda_j^{[x][2]}(t) = \sum_{i=1}^{N/n}(\lambda_j^{[i][1]}(t)\cdot p_{1to2}) \tag{4.16}$$

The transition probability from first to second stage p_{1to2} is taken as $\frac{1}{k}$ and Equation 4.16 becomes:

$$\lambda_j^{[x][2]}(t) = \frac{1}{k}\cdot\sum_{i=1}^{N/n}\lambda_j^{[i][1]}(t) \tag{4.17}$$

By using Equation 4.17 and Equation 4.13,

$$N_j^{[x][2]}(t) = \frac{\frac{(t_{MsOF}+(\frac{N}{n})^2\cdot t_r).\sum_{i=1}^{N/n}\lambda_j^{[i][1]}(t)}{k}}{1-\frac{(t_{MsOF}+(\frac{N}{n})^2\cdot t_r).\sum_{i=1}^{N/n}\lambda_j^{[i][1]}(t)}{k}} \tag{4.18}$$

For Stage$_3$, the arrival rate per $M/M/1$ system is:

$$\lambda_j^{[x][3]}(t) = \sum_{i=1}^{k}(\lambda_j^{[i][2]}(t)\cdot p_{2to3}) \tag{4.19}$$

The transition probability from second to third stage p_{2to3}, is taken as $\frac{n}{N}$:

$$\lambda_j^{[x][3]}(t) = \frac{n}{N} \cdot \sum_{i=1}^{k} \lambda_j^{[i][2]}(t) \tag{4.20}$$

By using Equation 4.20,

$$N_j^{[x][3]}(t) = \frac{\frac{(t_{MsOF}+(n.k).t_r).n.\sum_{i=1}^{k}\lambda_j^{[i][2]}(t)}{N}}{1-\frac{(t_{MsOF}+(n.k).t_r).n.\sum_{i=1}^{k}\lambda_j^{[i][2]}(t)}{N}} \tag{4.21}$$

However, to obtain total number of flows in a MsOF switch, Equation 4.14 should be simplified. Therefore, $\lambda_j^{[i][1]}(t)$ is set to n for all ts. According to this, $\lambda_j^{[i][2]}(t)$ equals to $\frac{1}{k}.\frac{N}{n}.n$, and also equals to $\frac{N}{k}$; and $\lambda_j^{[i][3]}(t)$ equals to $\frac{n}{N}.k.\frac{N}{k}$ that becomes n for all ts. Therefore, the total number of flows in OF switch is calculated by using equations 4.15, 4.18 and 4.21 implementations to 4.14:

$$\begin{aligned} N'_j(t) = 2.\frac{N}{n}.(\frac{n\cdot(t_{MsOF}+(n.k).t_r)}{1-n\cdot(t_{MsOF}+(n.k).t_r)}) \\ +k.(\frac{(t_{MsOF}+(\frac{N}{n})^2\cdot t_r).\frac{N}{k}}{1-(t_{MsOF}+(\frac{N}{n})^2\cdot t_r).\frac{N}{k}} \end{aligned} \tag{4.22}$$

According to Equation 4.4 and replacing $\lambda'_j(t)$ with $\sum_{i=1}^{\frac{N}{n}}\lambda_j^{[i][1]}(t)$ which equals to $\frac{N}{n}.n$ and also equals to N, the flow forwarding delay $T_{MsOF}(t)$ is obtain as follows:

$$\begin{aligned} T_{MsOF}(t) = T'_j(t) = \frac{2.(t_{MsOF}+n.k.t_r)}{1-n.(t_{MsOF}+n.k.t_r)} \\ +\frac{(t_{MsOF}+(\frac{N}{n})^2.t_r)}{1-(t_{MsOF}+(\frac{N}{n})^2.t_r).\frac{N}{k}} \end{aligned} \tag{4.23}$$

5. PERFORMANCE EVALUATION

In this section, proposed MsOF model and conventional OF switch model are compared according to spatial complexity parameter and temporal complexity parameter. The simulation environment is prepared with C++. There is a grid topology including one cell of 500x500 square meters. The simulation is repetaed for two cases: an MsOF switch located in the center of topology, and a conventional OF switch located in the center. For spatial complexity investigation, different flow table sizes ($n \cdot k$) and different number of input ports are considered for both of two cases. Similarly, for temporal complexity investigation, different arrival rates ($\lambda'_j(t)$) of flows are investigated. As a result, memory usage and flow forwarding delays are illustrated with some visual metarials for both MsOF switch and conventional OF switch in following sections.

5.1 Spatial Complexity

In Figure 5.1, results for MsOF switch hardware cost and conventional OF switch hardware cost (C_{MsOF} and C_{CON}), when individual table size($n \cdot k$) equals to 50, are observed according to decreasing number of input ports in the switches (N). In this figure, y-axis represents the memory usage in flow table pipeline of MsOF switch and conventional OF switch in Kbits. Similarly, x-axis represents five different topology scenarios with different number of input ports as 200, 160, 80, 40, and 20 for both MsOF and conventioanl OF switch cases. The values show different switch sizes with different number of input ports. Switch sizes are consdiered as the topology density that means if there is an urban area, then number of input ports is high. As seen in the figure, proposed MsOF switch has less spatial complexity than conventional one for large N values such as 40, 80, 160 and 200. Moreover, the memory usage gap between MsOF and conventional OF is opened as N increases that corresponds denser topologies with switches of large capacities. However, while $N = 20$, C_{CON} becomes less than C_{MsOF} because of individual table size used in both of the systems ($n \cdot k$) as

50. In this case, the topology corresponds to a rural area and a small conventional OF switch handles the incoming flows. However, our proposed MsOF architecture has multiple stages, 3-stages in this thesis, and has more flow tables than required for this rural topology. Thus, inner fragmentation of proposed MsOF architecture is more than the conventional one in this scenario and this leads to a redundant hardware cost.

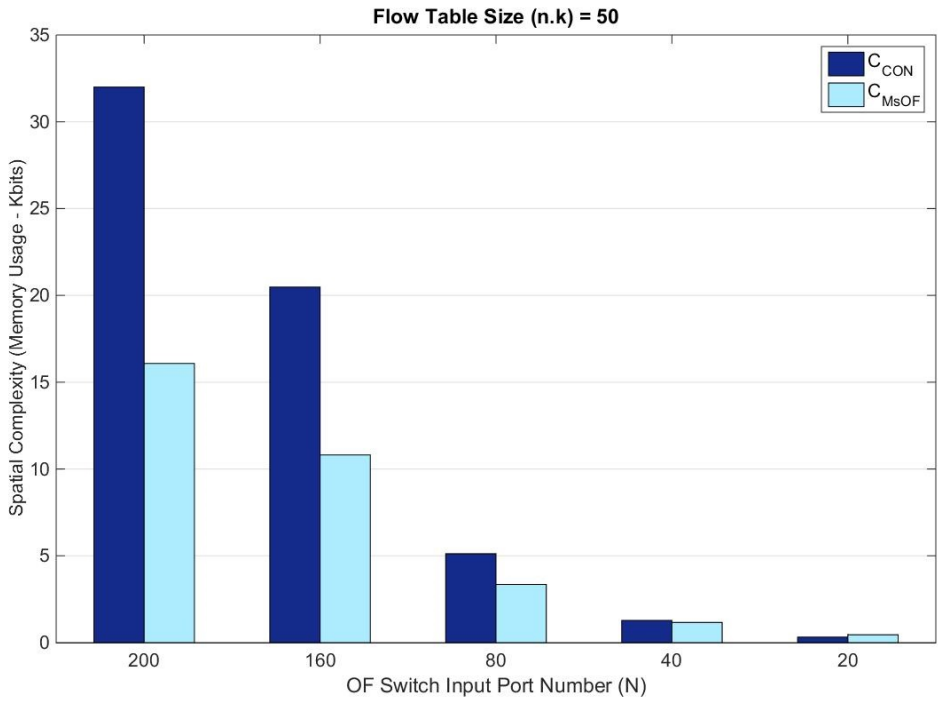

Figure 5.1: Comparison of C_{CON} and C_{MsOF} When each Table Size $n \cdot k$ Equals to 50.

Inner fragmentation of flow tables in the pipeline can be reduced by using smaller individual flow table sizes ($n \cdot k$). The scenarios with N equals to 40 and 20 in Figure 5.1 are investigated under 4 different schemes with 4 different individual flow table sizes ($n \cdot k$) as 50, 40, 30 and 20. Thereby, the effect of individual flow table size on inner fragmentation and also on spatial complexity is investigated. According to the simulation results, it can be asserted that when the table size becomes smaller, MsOF provides better spatial complexity even for small number of input ports, N, in OF switch. For example consider Figure 5.2, changing ($n \cdot k$) values in x-axis (which represents individual flow table sizes) from 50 to 20 decreased by 10, MsOF switch results in less cost of memory usage compared to conventional OF switch for different values of input ports number as 20 and 40. Moreover, as seen in the same figure, the gap between memory cost increases, showing that MsOF switch has much less spatial complexity than conventional OF switch, for the case of changing N from 40 to 20.

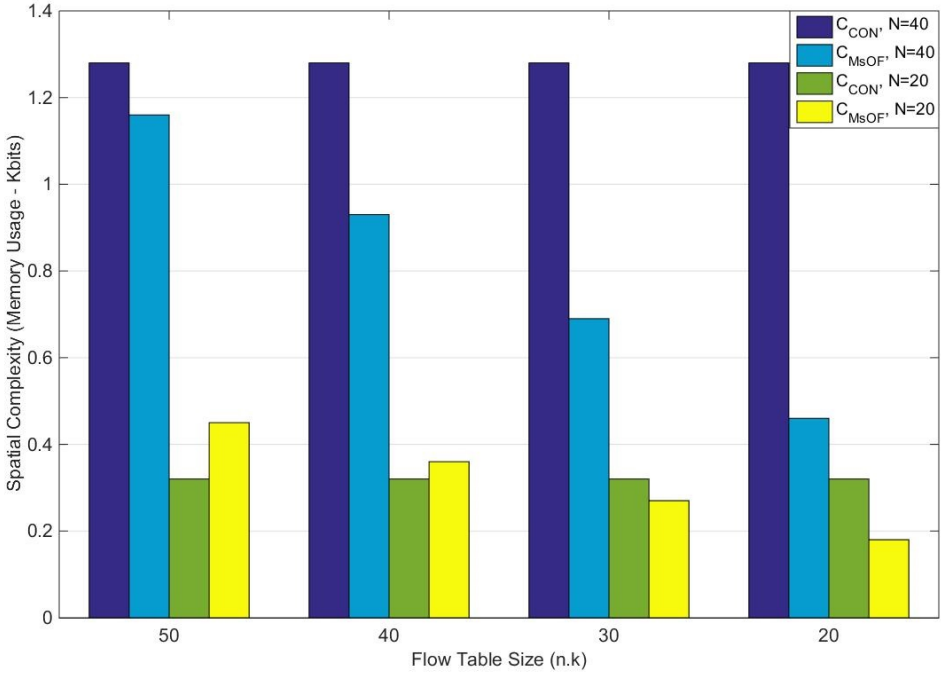

Figure 5.2: Comparison of C_{CON} and C_{MsOF} According to Different Table Sizes($n \cdot k$).

According to temporal complexity parameter used in this work, it can be asserted that MsOF switch provides less spatial complexity compared with conventional OF switch by using appropriate values of individual flow table sizes ($n \cdot k$) in the switch.

5.2 Temporal Complexity

The flow forwarding delay comparison between proposed MsOF switch and conventional OF switch is shown in Figure 5.3 according to total flow arrival rate ($\lambda'_j(t)$). There are 3 different scenarios considered in this simulation with 3 different number of input ports as 80, 40 and 20. These scenarios are indicated using 3 different legends in the graph and all of three scenarios are applied for both cases with MsOF switch and conventional OF switch in the topology. The same set of different total arrival rates are provided for the switch in the topology for all scenarios in the simulation. The arrival rate is also defined as network load in this work. X-axis is used to represent the normalized values of total traffic load by a percantage and called as network load. Y-axis of the graph repsents flow forwarding values in milliseconds under different traffic loads. At the light of these definitions, flow forwarding delay rises on both type of switches during the network load increases, as usual. However, MsOF switch always provides less delay than conventional OF switch

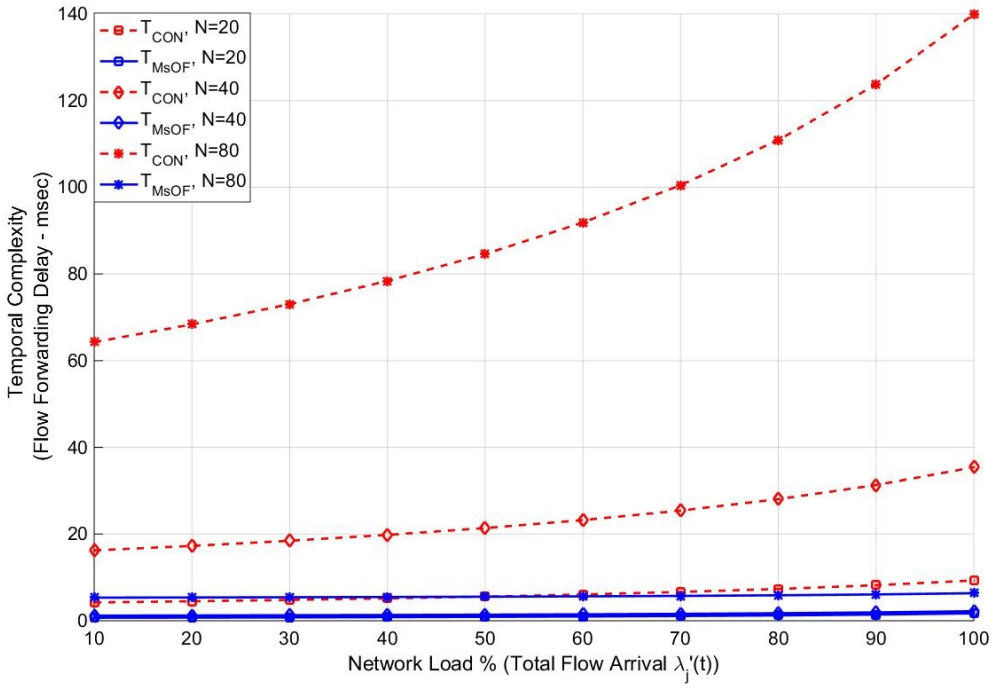

Figure 5.3: Comparison of T_{CON} and T_{MsOF} as the Percentage of Total Flow Arrival Rate($\lambda'_j(t)$) Increases.

for various input port numbers in the switches (N) as 20, 40 and 80. For example, MsOF forwarding delay is approximately 7 times less than the conventional one for the scenario that N equals 20 and network load is %60. Considering the proposed temporal complexity parameter in this work, it can be asserted that MsOF switch has less temporal complexity, i.e. flow forwarding delay than the conventional one according to simulation results.

6. CONCLUSIONS AND RECOMMENDATIONS

In this thesis, a novel Multi-stage OpenFlow (MsOF) switch model is proposed in order to eliminate negative effects of SDcN fashion in Data Plane according to spatial complexity and temporal complexity of convetional switch architecture, i.e. hardware cost and flow forwarding delay, respectively. A novel multi-stage architectural model is considered for flow table pipeline within the OF switch. A specific form of 3-stages is used for flow table interconnections. Moreover, advantages of NoC paradigm is inspired and adapted to proposed OF switch architecture by assigning an individual microprocessor for each flow table in the pipeline. Thus, parallel processing for incoming flows in flow table pipeline is achieved. With proposed MsOF switch model, flows can be forwarded with less memory usage and also less forwarding delay according to spatial-temporal complexity analysis. Moreover, dummy characteristic of OF switch paradigm is maintained with some little alteration in OF protocol headers. A new field named *next table identifier* is defined and suffixed to the existing header fields in order to indicate the next table in the pipeline.

According to performance analysis, redundant memory usage (redundant hardware cost of OF switch) is decreased and QoS of a flow is enhanced with the novel MsOF design. By using Queuing Theory, spatial and temporal complexity parameters for MsOF switch (C_{MsOF} and T_{MsOF}) are defined, examined and compared with conventional ones (C_{CON} and T_{CON}), respectively. Considering the performance results, MsOF has much less spatial and temporal complexity than conventional OF switch model, especially in dense areas. Using smaller individual table sizes (total of $n \cdot k$) in MsOF flow table pipeline, proposed architectural model provides less cost for memory usage to compare with conventional OF switch. Moreover, MsOF is able to forward flows with less delay compared to the one in conventional OF switch under urban areas. It is demonstrated in this thesis that the proposed multi-stage OF switch model has less spatio-temporal complexity compared to conventional OF

switch architecture considering spatial complexity parameter and temporal complexity parameter defined in this work.

As a future work, performance of proposed model will be investigated under different traffic loads in real network topologies with physical hardware components. This research can provide a basis for future works about SDcN paradigm, especially for works on Data Plane.

REFERENCES

[1] **McKeown, N.**, **Anderson, T.**, **Balakrishnan, H.**, **Parulkar, G.**, **Peterson, L.**, **Rexford, J.**, **Shenker, S. and Turner, J.** (2008). OpenFlow: Enabling Innovation in Campus Networks, *SIGCOMM Comput. Commun. Rev.*, ***38***(2), 69–74, `http://doi.acm.org/10.1145/1355734.1355746`.

[2] **ONF** (2011). OF Switch Specification Version 1.1.0, **Technical Report**, Open Networking Foundation.

[3] **ONF** (2013). OF Switch Specification Version 1.3.2., **Technical Report**, Open Networking Foundation.

[4] **Cisco** (2014). Visual Networking Index: Forecast and Methodology, 2013-2018, **Technical Report**, Cisco.

[5] **Kreutz, D.**, **Ramos, F.**, **Esteves Verissimo, P.**, **Esteve Rothenberg, C.**, **Azodolmolky, S. and Uhlig, S.** (2015). Software-Defined Networking: A Comprehensive Survey, *Proceedings of the IEEE*, ***103***(1), 14–76.

[6] **Gelberger, A.**, **Yemini, N. and Giladi, R.** (2013). Performance Analysis of Software-Defined Networking (SDN), *IEEE 21st International Symposium on Modeling, Analysis Simulation of Computer and Telecommunication Systems (MASCOTS)*, pp.389–393.

[7] **Zhou, S.**, **Jiang, W. and Prasanna, V.** (2014). A Flexible and Scalable High-Performance OpenFlow Switch on Heterogeneous SoC Platforms, *IEEE International Performance Computing and Communications Conference (IPCCC), 2014*, pp.1–8.

[8] **Tanyingyong, V.**, **Hidell, M. and Sjodin, P.** (2011). Using Hardware Classification to Improve PC-Based OpenFlow Switching, *IEEE 12th International Conference on High Performance Switching and Routing (HPSR), 2011*, pp.215–221.

[9] **Ahmed, R. and Boutaba, R.** (2014). Design Considerations for Managing Wide Area Software Defined Networks, *IEEE Communications Magazine*, ***52***(7), 116–123.

[10] **Bhuyan, L.**, **Iyer, R.**, **Wang, H.J. and Kumar, A.** (2000). Impact of CC-NUMA Memory Management Policies on the Application Performance of Multistage Switching Networks, *IEEE Transactions on Parallel and Distributed Systems*, ***11***(3), 230–246.

[11] **Piyachon, P. and Luo, Y.** (2006). Efficient Memory Utilization on Network Processors for Deep Packet Inspection, *ACM/IEEE Symposium on Architecture for Networking and Communications systems, 2006. ANCS 2006.*, pp.71–80.

[12] **Liu, M.**, **Ji, W.**, **Wang, Z.**, **Li, J. and Pu, X.** (2009). High Performance Memory Management for a Multi-core Architecture, *Computer and Information Technology, 2009. CIT '09. Ninth IEEE International Conference on*, volume 1, pp.63–68.

[13] **Iyer, R. and Bhuyan, L.** (2000). Design and Evaluation of a Switch Cache Architecture for CC-NUMA Multiprocessors, *IEEE Transactions on Computers*, ***49***(8), 779–797.

[14] **Kim, D.**, **Kim, M. and Sobelman, G.** (2006). DCOS: Cache Embedded Switch Architecture for Distributed Shared Memory Multiprocessor SoCs, *IEEE International Symposium on Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006*, pp.4 pp.–.

[15] **Akyildiz, I.**, **Lee, A.**, **Wang, P.**, **Luo, M. and Chou, W.** (2014). A Roadmap for Traffic Engineering in SDN-OpenFlow Networks, *Computer Networks: The International Journal of Computer and Telecommunications Networking*, ***71***, 1–30.

[16] **Antichi, G.**, **Di Pietro, A.**, **Giordano, S.**, **Procissi, G. and Ficara, D.** (2011). Design and Development of an OpenFlow Compliant Smart Gigabit Switch, *IEEE Global Telecommunications Conference (GLOBECOM 2011)*, pp.1–5.

[17] **Jin, H.**, **Pan, D.**, **Liu, J. and Pissinou, N.** (2013). OpenFlow-Based Flow-Level Bandwidth Provisioning for CICQ Switches, *IEEE Transactions on Computers*, ***62***(9), 1799–1812.

[18] **Altera** (2011). Applying the Benefits of Network on a Chip Architecture to FPGA System Design-WP-01149-1.1, **Technical Report**, Altera.

[19] **Arteris** (2005). A Comparison of Network-on-Chip and Busses, **Technical Report**, Arteris - The Network on Chip Company.

[20] **Bjerregaard, T. and Mahadevan, S.** (2006). A Survey of Research and Practices of Network-on-Chip, *ACM Computing Surveys*, ***38***(9), 1799–1812.

[21] **Forouzan, B.** (2006). *Data Communications and Networking*, McGraw-Hill, 4th edition.

[22] **ONF** (2013). OpenFlow Switch Specification Version 1.4.0, **Technical Report**, Open Network Foundation-ONF.

[23] **Erel, M.**, **Arslan, Z.**, **Özçevik, Y. and Canberk, B.** (2015). Grade of Service (GoS) based adaptive flow management for Software Defined Heterogeneous Networks (SDHetN), *Computer Networks*, ***76***(0), 317 – 330, `http://www.sciencedirect.com/science/article/pii/S1389128614004010`.

[24] **Sünnen, D.** (2011). Performance Evaluation of OpenFlow Switches, *Semester Thesis at the Department of Information Technology and Electrical Engineering*, pp.186–191.

[25] **Yeganeh, S.**, **Tootoonchian, A. and Ganjali, Y.** (2013). On Scalability of Software-Defined Networking, *IEEE, Communications Magazine*, ***51***(2), 136–141.

[26] **Feng, T.**, **Bi, J. and Hu, H.** (2011). OpenRouter: OpenFlow Extension and Implementation Based on a Commercial Router, *19th IEEE International Conference on Network Protocols (ICNP), 2011*, pp.141–142.

[27] **Dely, P.**, **Vestin, J.**, **Kassler, A.**, **Bayer, N.**, **Einsiedler, H. and Peylo, C.** (2012). CloudMAC 2014; An OpenFlow Based Architecture for 802.11 MAC Layer Processing in the Cloud, *2012 IEEE, Globecom Workshops (GC Wkshps)*, pp.186–191.

[28] **Gross, D.**, **Shortle, J.**, **Thompson, J. and Harris, C.** (2008). *Fundamentals of Queueing Theory*, Wiley, 4th edition.

CURRICULUM VITAE

Name Surname: Yusuf ÖZÇEVİK

Place and Date of Birth: İzmir - 11/02/1991

Adress: Istanbul Technical University, Ayazağa Campus, Former Rectorate Building Maslak-Sarıyer/İstanbul

E-Mail: ozcevik@itu.edu.tr

B.Sc.: Istanbul Technical University - Computer Engineering

Academic Services:

- **TPC Member** - The International Workshop on Energy Awareness for Heterogeneous Networks: Recent Hardware and Software-Defined Solutions (EAHN 2014), in conjunction with IEEE DCOSS 2014
- **Reviewer for Journal** - The International Journal of Communication Systems (IJCS), 2014-present
- **Reviewer for Conference** - IEEE International Black Sea Conference on Communications and Networking (BlackSeaCom), May 2014

Professional Experience and Rewards:

- Senior Software Developer at Istanbul Technical University Computing Department since February, 2014
- BSc Graduation Ranking in Faculty: 1st
- High School Graduation Ranking: 1st

List of Publications:

- **Y. Ozcevik**, M. Erel, B. Canberk, 'Spatio-Temporal Multi-stage OpenFlow Switch Model for Software Defined Cellular Networks', in **IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)**, Boston-USA, September 2015.
- **Y. Ozcevik**, M. Erel, B. Canberk, 'NOC based Banyan OpenFlow Switch for Software Defined Networks', in **IEEE 23. Signal Processing and Communications Applications Conference**, Malatya-TURKEY, May 2015.
- M. Erel, Z. Arslan, **Y. Ozcevik**, B. Canberk, 'Software-Defined Wireless Networking: A New Paradigm for Next Generation Network Management Framework', Modeling and Simulation of Computer Networks and Systems: Methodologies and Applications, Edited by M.S. Obaidat, F. Zarai and P. Nicopolitidis, **ELSEVIER PUBLICATIONS**, 2014.

▪ M. Erel, Z. Arslan, **Y. Ozcevik** and B. Canberk (2015). 'Grade of Service (GoS) based adaptive flow management for Software Defined Heterogeneous Networks (SDHetN)', **Computer Networks - ELSEVIER PUBLICATIONS**, 76(0), 317 – 330.

▪ Z. Arslan, M. Erel, **Y. Ozcevik**, B. Canberk, 'SDoff: A Software-Defined Offloading Controller for Heterogeneous Networks', in **IEEE Wireless Communications and Networking Conference (IEEE WCNC)**, Istanbul-TURKEY, April 2014.

▪ M. Erel, **Y. Ozcevik**, B. Canberk, 'A Topology Control Mechanism for Cognitive Smallcell Networks Under Heterogeneous Traffic', in **IEEE 14th International Symposium and Workshops on a World of Wireless, Mobile and Multimedia Networks (IEEE WoWMoM)** , Madrid-SPAIN, June 2013.

PUBLICATIONS ON THE THESIS

▪ **Y. Ozcevik**, M. Erel, B. Canberk, 'Spatio-Temporal Multi-stage OpenFlow Switch Model for Software Defined Cellular Networks', in **IEEE 82nd Vehicular Technology Conference (VTC2015-Fall)**, Boston-USA, September 2015.

▪ **Y. Ozcevik**, M. Erel, B. Canberk, 'NOC based Banyan OpenFlow Switch for Software Defined Networks', in **IEEE 23. Signal Processing and Communications Applications Conference**, Malatya-TURKEY, May 2015.