Optimization of the OpenFlow Controller in Wireless Environments for Enhancing Mobility

Abdallah AL Sabbagh, Pakawat Pupatwibul, Ameen Banjar and Robin Braun
Centre for Real-Time Information Networks (CRIN)
University of Technology, Sydney (UTS)
Sydney, Australia
asabbagh@eng.uts.edu.au, ndomon66@gmail.com, dr.banjar@gmail.com, robin.braun@uts.edu.au

***Abstract*—OpenRoads or OpenFlow Wireless is an open-source platform for deploying an innovative and realistic strategy for different services in wireless networks. It provides a wireless extension for OpenFlow. It is developed to support existing Wireless Local Area Networks (WLANs) and Worldwide Interoperability for Microwave Access (WiMAX) networks. It can provide several mobility managers and run them concurrently in the network including *hard handover*, *informed handover*, *n-casting* and *Hoolock*. However, the provided mobility support for flow-based routing, where flows of one source taking different paths through multiple wireless access points or base stations, is not simple and hard to be deployed in the traditional routing algorithms. This paper proposes an intelligent mobility enhancement control and then develops an algorithm to decide which neighbor switches need to be selected for the installation of new flow entries and to allocate the appropriate *idle-timeout* for the selected switches. The proposed approach provides a simple solution to solve the user mobility problem in wireless OpenFlow environments which can handle the fast migration of user addresses (e.g. IP addresses) between several wireless access points and base stations. This approach leads to improvement in the end users' experience.**

***Index Terms*—OpenRoads; OpenFlow Wireless; User Mobility; Vertical Handover; Next Generation Networks.**

I. INTRODUCTION

Today's mobile applications and the advancement of wireless technologies, such as Wireless Sensor Networks (WSN) and Next Generation Wireless Networks (NGWN) are expected to be heterogeneous [1]. For example, Long Term Evolution (LTE) and Mobile Worldwide Interoperability for Microwave Access (WiMAX) may lead to many problems. Firstly, they are inflexible to policy changes; this is because of dynamically changing business needs make policy changes complex and difficult to cope with in wireless networks. Applying changed algorithms often require direct manual re-configuration or re-programming [2]. This can incur delay in policy enforcement. Secondly, transmitted and received packets in mobile networks are hard to manage. This is because when a device moves to a new subnet it needs a new Internet Protocol (IP) address and all outstanding connections are lost. Solutions currently applied in Mobile IP are difficult to implement.

OpenFlow is an emerging network paradigm which decouples the control path from the data path of forwarding elements. Using this strategy, OpenFlow allows network element devices such as routers and switches to be programmable. In addition, OpenFlow provides high flexibility of novel packet forwarding and routing of network flows that can adapt to the ever changing environment by using virtualization and flow-based routing [3]. Several approaches for routing and forwarding in Wireless OpenFlow networks have been proposed in the literature to enable client mobility and optimized solutions for mobility management, such as OpenRoads.

This paper describes how an OpenFlow network provides mobility across different wireless networks by considering three different scenarios where several Mobile Terminals (MTs) traverse between different access points and base stations. The MTs are running software that is connected to a controller which is running inside a virtual machine. However, when a user moves to a different access point or a base station, usually it will require a new subnet and a new IP address and all outstanding flows need to be updated and modified. This could introduce undesired delay to the client, harming the user experience.

To address this issue, the paper proposes a mobility enhancement control that optimizes the OpenFlow controller and provides flexible control, virtualization and high-level abstraction flow-based routing and forwarding capabilities for mobile users. The proposed approach aims to solve the user mobility problem in wireless OpenFlow networks which manages the fast movement of MT addresses (e.g. IP addresses) across OpenFlow access points/base stations, and thus significantly reduces the interaction between the switch and controller.

The proposed mobility management demonstrates the feasibility of our approach which leads to promote one way of decision making and mobility mechanism and to show how our switch selection algorithm can offer new mobility managers to be quickly deployed and prototyped by other researches.

In this paper, the user's mobility in wireless OpenFlow networks is presented. The main contribution of this paper is the designing of an intelligent algorithm that allows for flexibility and efficient use of packet processing. By implementing the proposed approach, controllers can be optimized as well as maintaining all existing connections seamlessly. Moreover, OpenFlow controllers will have ultimate control over packet paths that enable clients to seamlessly roam between different access points and base stations in order to improve the users' experience.

The rest of the paper is outlined as follows. Section II presents the key fundamental details of OpenFlow network. In Section III, related work on mobility management in wireless OpenFlow and on OpenFlow applications is described. Section IV evaluates different scenarios to handle user mobility in wireless OpenFlow. In Section V, the proposed algorithm for enhancing mobility is shown. A mobility manager application for the proposed algorithm is also presented in this section. Finally, Section VI summarizes and concludes the paper by providing directions for future work.

II. OpenFlow Network

OpenFlow platform is an open standard that enables new opportunities to realize rich network control mechanisms and experiments by allowing researchers to flexibly program control path functionalities on the controller in production networks. The OpenFlow architecture separates the high level routing decision making from the forwarding elements which make it possible to run in software and to be programmable [4]. Therefore, the OpenFlow switch becomes simple. It only implements the data path packet switching functions. Whereas a specific node, the controller manages the control path functionalities and centralizes the execution of all decision tasks by installing, deleting, and modifying flow entries on the OpenFlow switches. The OpenFlow protocol is used to communicate between the OpenFlow switch and controller. This protocol defines messages such as packets received, send packets out, modify forwarding tables, and get stats [5].

OpenFlow switch consists of several Flow-Tables, and every one of them includes multiple flow entries. A flow entry in the Flow-Table contains three fields. Firstly, a set of packet headers that specifies the flow. It is used for matching packets against flow entries. Then, counters which updates the statistics such as duration of time the flow entry has been installed in the switch and the number of received and transmitted packets for each flow. Finally, an associated action which is implemented to specify the procedure of the packets such as forwarding out a port, modifying field, or to drop [6]. Whenever an OpenFlow switch receives a flow's first packet, in which it has never seen before nor has a matching flow entries, that first packet will be transmitted to the controller. The first packet is called a "*Flow Request*". The controller then makes a decision and computes a path for this flow, and installs flow entries on every switch throughout that chosen path. Ultimately, the packet itself will be forwarded back to the origin switch from the controller and sent to the destination.

Fig. 1 shows the sequence processes of OpenFlow packet processing. In this example, Host 1 would like to communicate and send packets to Host 2. When the first packet reaches its connected switch (Switch 1), it will be forwarded by the switch to a controller which will analyze the packet and then create new flow entries for this packet. These flow entries will be installed by the controller on the switches within the chosen path (Switch 1, Switch 2 and Switch 3). Then the controller will return the packet back to Switch1 from which it will be transmitted through the chosen path to reach its destination host (Host 2).

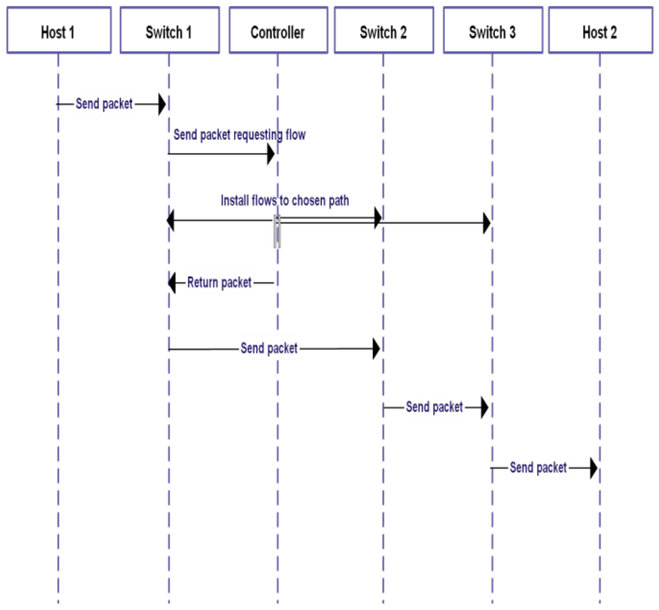

Fig. 1. OpenFlow packet processing.

III. Related Work

This section presents a literature review on OpenFlow. The wireless OpenFlow and the OpenFlow applications are described in the following subsections.

A. Wireless OpenFlow

OpenRoads is an innovative open source platform for mobile networks. It creates a way for researchers to run multiple network experiments using their own production traffic built and deployed on a wireless extension to OpenFlow. Thus, OpenRoads can be considered as "OpenFlow Wireless" [7].

The OpenRoads' structure includes three various layers: flow, network slicing and controller. All of these layers can provide high level abstraction, virtualization, and flexible control. This division will allow researchers to construct widely different algorithms and test them at the same time in a production wireless traffic. OpenRoads also incorporates many wireless access technologies such as Wireless Local Area Network (WLAN) and WiMAX.

In ref. [8, 9], OpenRoads run experiments to show the loss-less handover between the WLAN and WiMAX nodes. They achieve this by redirecting the traffic flows dynamically through the network. OpenRoads has the capability to provide very efficient control of mobility policies such as break-before-make and make-before-break connections. The isolation capabilities of FlowVisor for OpenRoads are tested by the heavy use of “read-only” FlowSpace [10].

The OpenRoads deployment has been used to support mobility management in OpenFlow networks. However, OpenRoads does not involve the integration of existing routing protocols and multi-hop wireless communications.

Ref. [3] proposed another approach which aims to enable client mobility solution by extending the OpenFlow infrastructure and develop a simple application for mobility management. However, the goal of that study did not give an optimized solution for mobility management in wireless OpenFlow. It only shows that few lines of codes in Python can be implemented as a reasonable and useful service.

B. OpenFlow Applications

A large number of OpenFlow applications have been developed, such as Maestro, to solve the limitations of OpenFlow control plane. Maestro has a central controller for a flow-based routing network, with increasing flow processing; this will offer scalability. Maestro can achieve scalability by coordinating between centralized controllers and distributed routing protocols. The Maestro approach works as a hybrid control plane, which is more robust than the centralized control plane. It keeps the simple programming model for programmers, and uses the technique of parallelism. This includes the use of multi-threading to handle the flow requests from OpenFlow switches and batches sending when the controller needs to send control configuration messages back to the switches. Maestro can solve traffic bottlenecks by using several applications, such as “routing” or “learning switch” [11].

Maestro has a programmable environment with a high-level language, which can deal with distribution and concurrency without the need to involve the operator [12]. Maestro has a user interface to control the hardware; it also includes analysis tools [12]. It can auto-detect the attached hardware including its features. Moreover, Maestro can make local decisions without involving a master, and can synchronize certain actions by using the master [13]. The operator is not involved in the controller configuration, where Maestro is able to insert events. As a result, Maestro has been accepted as high-level program applications for protected structures [11].

IV. OpenFlow in Mobility

A mobile wireless platform based on OpenFlow was developed typically for experimental research and realistic deployment of network and services. In this section, the ease of developing mobility services on top of OpenFlow wireless to control flows in the network as well to improve end user experience is demonstrated. In addition, this section shows how wireless MT can seamlessly handover between different wireless networks and technologies in OpenFlow. Fig. 2 illustrates the network topology consisting of multiple OpenFlow switches, WLAN access points and other wireless base station such as 3G and 4G base stations. These devices are controlled by a central controller where all the high level network abstractions are implemented. The transmission of packets from a MT host to an end host user are detailed in three different cases with multiple wireless interfaces [14].

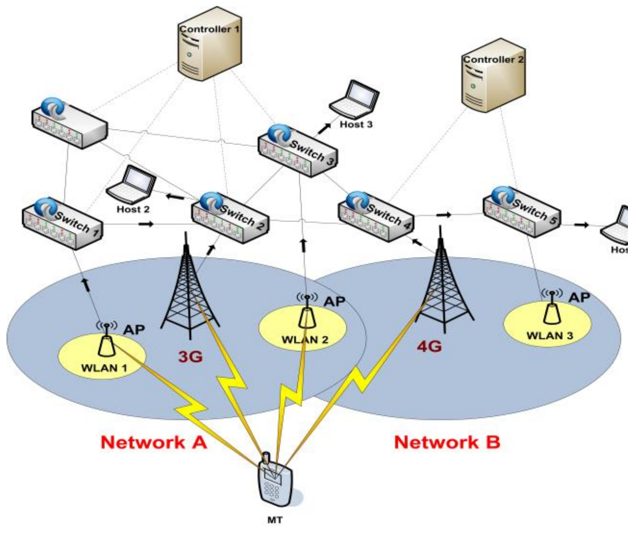

Fig. 2. User mobility in multi-access network.

Fig. 3 presents a simplified sequence diagram to show how the packet is being routed to the end host user.

In Case 1, when the MT moves and performs handover in the network, the flows can be rerouted with ease across different WLANs and base stations. If the flows traverse to a new switch that already has similar flows installed earlier because of the same chosen path, for example, as the MT moves while streaming a video and performs handover from the WLAN 1 to the 3G base station, it does not require to send the first packet again to the controller for routing decisions. In this case, the new switch will match the flow entry to the flow table and executes the associated actions.

In Case 2, when the MT moves and performs handover to another WLAN access point or base station that is connected to a switch which is not in the selected path, because there is no flow entry configured in this switch’s flow table, for example, as the user moves from the 3G base station to the WLAN 2, the first packet will be sent to the controller. The controller then has to calculate the path for this flow and install them again to a new selected path. Finally, the packet will be returned the switch from the controller and will be forwarded to the destination through to the specified port.

In Case 3, when the MT moves and performs handover to a WLAN access point or base station that is in a different network with another controller, for example as the MT moves from the WLAN 2 to the 4G base station, the first packet will be forwarded to the new controller because it has never seen the packet before. Then, the normal packet processing will be repeated again, where the controller will have to determine the route and install the flow entries to every switch along the chosen path. Finally, the packet will be sent from the connected switch to the end host user.

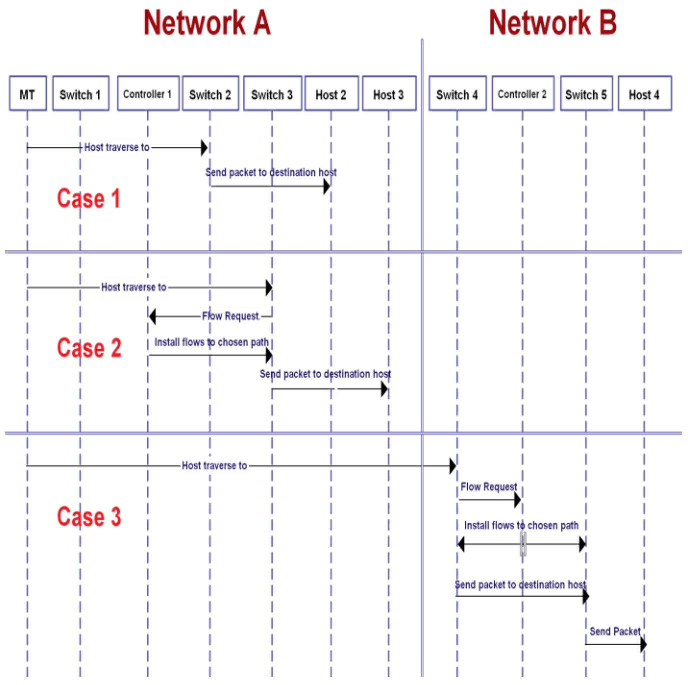

Fig. 3. OpenFlow packet processing in user mobility.

The section has been presented a MT user which can exploit multiple access networks of similar and different wireless technologies to improve end user experience while maintaining seamless pervasive connectivity. This is hoping to inspire others to come forth to create the future of mobile wireless network and validate their proposals of wireless OpenFlow networks.

V. An Intelligent Algorithm for Enhancing Mobility

OpenRoads is developed to support wireless technologies and user mobility in OpenFlow environment. However, OpenRoads does not provide an efficient solution for mobility. In this section, we propose an intelligent algorithm to enhance mobility for MT in OpenFlow networks. The proposed intelligent algorithm for enhancing mobility in OpenFlow and the mobility manager application for the proposed algorithm are described in the following subsections.

A. Intelligent Algorithm for Enhancing Mobility

OpenFlow network consists of several switches and a controller which is responsible for managing flow entries within these switches. Whenever a new flow request has been received from the OpenFlow switch, the controller needs to determine the network rules and distributes appropriate action on how to handle packets within the network devices. The controller has also the role of adding, deleting, and modifying the flows and installing them to the associated switches.

In this subsection, we propose an intelligent algorithm to enhance mobility for user terminal in OpenFlow. In the proposed algorithm, the flow entry will not only be created and installed into the associated switches. It will also be created and installed into selected neighbor switches to accommodate the MTs. These installations will reduce the processing time for the neighbor switches in case of a MT has been moved to any of the access points or base station within these switches. A mobility manager application is developed to decide which neighbor switches will be selected for the installation of new flow entries. Fig. 4 shows the flowchart for the proposed mobility manager algorithm.

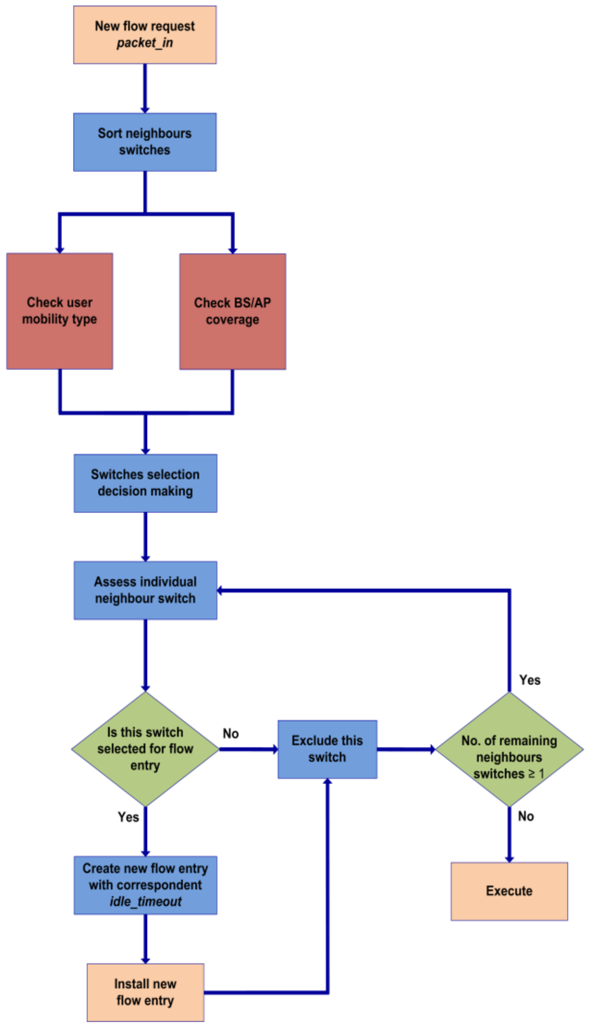
New flow request
packet_in
Sort neighbours switches
Check user mobility type
Check BS/AP coverage
Switches selection decision making
Assess individual neighbour switch
Is this switch selected for flow entry
No
Exclude this switch
No. of remaining neighbours switches ≥ 1
Yes
Yes
No
Create new flow entry with correspondent idle_timeout
Execute
Install new flow entry

Fig. 4. Proposed intelligent algorithm for enhancing mobility.

When a new flow request is received, the controller will first evaluate the received request, and then create and install flow entries for the chosen path switches. Next, the controller will make a list of available neighbor switches. After that, the mobility manager application will collect information about the MT such as velocity type (low, medium or high) and wireless access points/base stations network coverage range that belong to the neighbor switches. After that, a decision will be made by the controller to select the appropriate neighbor switches where flow entries need to be created and installed and to allocate the appropriate *idle-timeout* that needs to be created for each flow entry. For example, if the MT is in low mobility, the allocated *idle-timeout* will be longer for the installation of flow entries in the further distance switches. The created flow entries for packets transmitted by high mobility users will have a shorter *idle-timeout* that needs to be installed in the near switches. Then, flow entries will be created with different *idle-timeout* for the associated and selected neighbor switches. Finally, the created flow entries will be installed into the selected switches and then the packet will be forwarded by the transmitted switch.

B. Mobility Manager Application

The mobility manager application is presented in this subsection. More details on the algorithm of this application are shown below. The OpenFlow controller is enhanced to install necessary flow entries for the selected neighboring OpenFlow switches by analyzing the user mobility type and the connected neighbors access points/base stations coverage range.

Algorithm 1 Mobility Manager Application's Algorithm

```
for pr ϵ 0 ... pmax (where pr is packet request sent by
OpenFlow switch to the controller)
  if packet is transmitted by a MT
      create flow entries;
      install flow entries to chosen path switches;
      sort neighbor switches that are not in the chosen path;
      check user mobility type;
      check network coverage range for access points and
            base stations that belong to the neighbor switches;
    for  all neighbor switches
      if neighbor switch has high coverage range
        if MT is in high mobility
            create flow entry with short idle_timeout;
            install flow entry to this switch;
        else
            create flow entry with long idle_timeout;
            install flow entry to this switch;
        end if
      else
        if MT is in high mobility
            create flow entry with short idle_timeout;
            install flow entry to this switch;
        else
            deny creation and installation  of flow entry
                  for this switch;
        end if
      end if
    end for
  else
      create flow entries;
      install flow entries to chosen path switches;
  end if
end for
```

VI. Conclusion

This paper proposes an intelligent algorithm to enhance mobility control in wireless OpenFlow environments. The proposed approach aims to support the fast migration of MT addresses across several OpenFlow access points and base stations. It also aims to reduce the interaction between the switch and controller in order to improve the user's experience. The paper also describes the concepts of current OpenFlow networks and shows how OpenRoads can provide mobility across different wireless networks referring to the three different cases. The proposed mobility solution will hopefully enhance the user mobility by determining the proper neighbouring switches that need installation for new flow entries. In addition, the proposed approach will allocate different appropriate *idle-timeout* for the selected switches by considering the user mobility type and the access point/base station coverage which will provide efficient control. Ultimately, it is expected to be the inspiration of a new generation of mobile wireless networks which are flexible and easy to maintain and it may even stimulate interesting proposals from the OpenFlow community showing a wide range of innovation opportunities. Future work can be an implementation of the designed algorithm and a run of experiments using OpenRoads by specially focusing on forwarding capabilities.

Acknowledgment

This work is sponsored by the Centre for Real-Time Information Networks (CRIN) in the Faculty of Engineering & Information Technology at the University of Technology, Sydney (UTS).

References

[1] A. AL Sabbagh, R. Braun and M. Abolhasan, "A Comprehensive Survey on RAT Selection Algorithms for Heterogeneous Networks," *World Academy of Science, Engineering and Technology (WASET)*, no. 73, January 2011, pp. 141-145.

[2] T. Luo, H. Tan and T. Quek, "Sensor OpenFlow: Enabling software-defined wireless sensor networks," *IEEE Communications Letters*, vol. 16. no. 11, November 2012, pp. 1896-1899.

[3] P. Dely, A. Kassler and N. Bayer, "Openflow for wireless mesh networks," *20th International Conference on Computer Communications and Networks (ICCCN 2011)*, Maui, Hawaii, July 31 - August 4, 2011, pp. 1-6.

[4] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker and J. Turner, "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, April 2008, pp. 69-74.

[5] Open Networking Foundation, "Openflow Switch Specification," version 1.3.0 (Wire Protocol 0x04), September 2012.

[6] Open Networking Foundation, "Software-Defined Networking: The New Norm for Networks," ONF White Paper, April 2012.

[7] K. K. Yap, M. Kobayashi, R. Sherwood, T. Y. Huang, M. Chan, N. Handigol and N. McKeown, "OpenRoads: Empowering

research in mobile networks," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, 2010, pp. 125-126.

[8] K. K. Yap, T. Y. Huang, M. Kobayashi, M. Chan, R. Sherwood, G. Parulkar and N. McKeown, "Lossless Handover with n-casting between WiFi-WiMAX on OpenRoads," *ACM Mobicom (Demo)*, 2009

[9] R. Sherwood, M. Chan, A. Covington, G. Gibb, M. Flajslik, N. Handigol, T. Y. Huang, P. Kazemian, M. Kobayashi and J. Naous, "Carving research slices out of your production networks with OpenFlow," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 1, 2010, pp. 129-130.

[10] D. Erickson, G. Gibb, B. Heller, D. Underhill, J. Naous, G. Appenzeller, G. Parulkar, N. McKeown, M. Rosenblum and M. Lam, "A demonstration of virtual machine mobility in an OpenFlow network," *Proceedings of ACM SIGCOMM (Demo)*, 2008, p. 513.

[11] Z. Cai, A. L. Cox, and T. E. N. Maestro, "Maestro: A system for scalable openflow control," *Technical Report TR10-08*, Rice University, Tech. Rep., 2010.

[12] S. Kandula, S. Sengupta, A. Greenberg, P. Patel and R. Chaiken, "The nature of data center traffic: measurements & analysis," *Proceedings of the 9th ACM SIGCOMM conference on Internet measurement conference*, ACM, 2009, pp. 202-208.

[13] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel and S. Sengupta, "VL2: a scalable and flexible data center network," *ACM SIGCOMM Computer Communication Review*, vol. 39, ACM, 2009, pp. 51-62.

[14] K. K. Yap, M. Kobayashi, R. Sherwood, N. Handigol, T. Y. Huang, M. Chan and N. McKeown, "Towards Rapid Innovation in Mobile Networks," OpenFlow Switching, June 2009.