Automated Controllers for Bandwidth Allocation in Network Virtualization

M. Said Seddiki, Bilel Nefzi, Ye-Qiong Song, Mounir Frikha

Automated Controllers for Bandwidth Allocation in Network Virtualization

M. Said Seddiki[†*], Bilel Nefzi[*], Ye-Qiong Song[*], Mounir Frikha[†]
[*]Higher School of Communications of Tunis, University of Carthage, Tunisia
[†]LORIA Research Laboratory, University of Lorraine, France
[†]{seddiki.said, m.frikha} @supcom.rnu.tn ; [*]{bilel.nefzi, ye-qiong.song} @loria.fr

***Abstract*—The concept of network virtualization was introduced to facilitate flexible service deployment for the future Internet. This recent technology provides a powerful tool to run multiple logical networks on the same physical substrate defined as virtual networks (VNs). Each physical link is split into virtual links and each VN receives a fraction of the available capacity. Bandwidth allocation for multiple VMs aims at sharing the physical links among multiple VNs. It is a critical challenge for both service providers (SPs) and infrastructure providers (InPs). This allocation should take into account the quality of service (QoS) requirements of the flows that are crossing each VN. In this paper, we consider a virtualized network environment where the SP deploys multiple VNs with different links' capacity demands and QoS requirements. Each VN competes with other VNs to receive fractions of physical links managed by multiple InPs. We present a two-layer controller system that adapts to the dynamic change of the workload of each VN. The system uses a prediction-based approach in order to find the optimal request for each VN. The request depends on the estimation of the relationship between the VN performance in terms of packet delays and the actual and past allocations. Then, due to the capacity constraint of the physical link, the system adjusts the offered bandwidth for each of them. Our model offers flexible distributed autonomous control of the bandwidth allocation to maintain the offered QoS to each VN at the desired level in response to the dynamics of the workload. Our mechanism provides an optimum allocation of the physical links by distributing the bandwidth periodically. It also offers the possibility of adjusting the VNs' parameters to take into account the current network behaviour to avoid bottleneck virtual links.**

***Keywords*-network virtualization; dynamic bandwidth allocation; feedback controller.**

I. Introduction

Over the last decades, Internet has seen exponential growth, and deploying new services has become more and more difficult. These services have stringent delay requirements that the current Internet architecture cannot provide and maintain. Network virtualization was introduced as a promising strategy for addressing this problem. This technology allows multiple logical networks to coexist on a shared physical substrate infrastructure [1]. The basic entity in a network virtualization environment is a virtual network (VN). It is a logical topology composed of a set of virtual nodes and links. The concept of network virtualization divides the role of the Internet service provider (ISP) into two separate entities; the service provider (SP) and the infrastructure provider (InP) [2].

A critical issue in network virtualization is virtual network embedding (VNE) [3]. This deals with the allocation of the physical nodes and links which consists of finding efficient and optimal mapping of virtual nodes and virtual links onto the substrate network resources. Virtual network embedding comprises three steps [4]. The first is the resource discovery, where each InP monitors its physical network and shares information with multiple SPs about the load, usage and performance of the substrate network using some measurement processes [5]. The second step is the virtual network mapping. This step is performed by the SP to match its requests with the available network resources. It is seen as the most complex step, because there is a need to combine both node and link constraints [6]. The last step is the virtual network allocation. This is the process of reserving and allocating physical resources to elements such as virtual nodes and virtual links. This task is performed by the InP upon the receipt of all the requests from the SPs. The existing approaches can be categorized as centralized or distributed approaches in a static or dynamic way [7]. Some solutions, to address this issue, solve a specific task of the embedding problem, while others are hybrids of two tasks, such as resource discovery and network mapping.

Bandwidth allocation is part of the virtual network allocation. It has the objective of fairly and efficiently sharing the physical links among multiple VNs. The challenge is to improve the utilization of the network resource and avoid congestion in the physical network [8]. Since this problem is considered an NP-hard problem, many heuristic algorithms have been proposed to address this issue [9].

In this paper, we aim to provide a fine-grained bandwidth allocation mechanism for concurrent active VNs on top of a physical network. Due to the dynamic workloads that are crossing each of them, this task can be seen as an optimization problem involving constraints such as the available physical capacity and the QoS requirements of each VN. The goal of this work is to offer an adaptive dynamic bandwidth allocation between multiple VNs by presenting a system architecture with multiple controllers. These controllers aim to find the optimal capacity fraction to request and to allocate for each VN. We suppose that the behaviour of each VN can be locally approximated, at a single instant, by a linear model that is a relationship between its past and present allocations and its

* This work has been partially supported by ANR-NSFC Quasimodo project (under No. ANR 2010 INTB 0206 01)

978-1-4799-3214-6/13/$31.00 ©2013 IEEE

past performances in terms of packet delays.

This paper is organized as follows. In section II, we expose the background and related work. In section III, we describe the system architecture model and present the approach adopted by each controller and propose an algorithm for dynamic resource allocation for multiple VNs. In section IV, we present and discuss the simulation results of our approach. Finally, in section V, we conclude and summarize our findings.

II. Background and related work

Botero and Hesselbach [10] studied the problem of bandwidth allocation among VNs and presented several mechanisms to offer a fair bandwidth distribution. The authors propose the utilization of fair mechanisms in order to solve the bandwidth distribution problem in virtual networking. They suggested distributing the bandwidth among competing VNs to avoid the strangulation of the VNs. Most of the proposed mechanisms do not offer any dynamic allocation of the link capacity between multiple VNs. They lead to a significant under-utilization of the available physical network resources.
Rahman et al. [11] formulated a VN embedding (SVNE) problem and presented an efficient heuristic for solving it with the assumption that the InP network does not remain operational at all times. The authors suggested a fast re-routing strategy and utilizing a pre-reserved quota for backup on each physical link. The proposed solution deal with link failures and supports node migration.
He et al. [12] proposed a flexible architecture called DaVinci that supports dynamic and adaptive bandwidth allocation for multiple VNs. The proposed method uses optimization theory to maximize the aggregate performance across the VNs. Each physical link periodically reassigns bandwidth fractions based on local link loads between its virtual links, while each virtual network runs its own traffic-management protocols that maximize its own performance objective independently. The authors showed that in the proposed architecture, the bandwidth shares converge quickly to the optimal values. The proposed solution did not consider the scenario when all the VNs become greedy and ask for more and more bandwidth capacity.

Economic models and game-theoretic methods have been offered to solve the optimization problem from the viewpoint of the InPs and the SPs. Various solutions have been proposed to address the problem of players maximizing their returns that depend on actions of other players. Zhou et al. [13] developed a non-cooperative game model for bandwidth allocation in the network virtualization environment, where the total bandwidth requirements of multiple VNs exceed the capacity of the physical network. In the proposed model, the InPs play the role of forcing VNs to modify their strategies in the form of a pricing scheme. The authors proposed an iterative algorithm to achieve the Nash Equilibrium. This model only focuses on how InPs allocate the limited bandwidth among multiple VNs, where a single SP can only obtain physical resources from a single InP.
Wang et al. [14] presented a Stackelberg game-theoretic model for dynamic bandwidth allocation, where VNs are the competing players and the result is the efficient and fair distribution of link capacity. At the upper level, the VNs that are the followers play a non-cooperative bandwidth allocation game, while at the lower level the substrate, which is the leader, sets a price to drive VNs to maximize revenue and to maximize social welfare. The proposed model maximizes the revenue of both InPs and SPs and proves the existence of a unique Stackelberg equilibrium point. However, the authors did not take into account the QoS requirements of the flows that are crossing each VN.
Seddiki et al. [15], presented an approach based on two-stage non-cooperative games for bandwidth allocation that aims at reducing the complexity of network management and avoiding bandwidth performance problems in a virtualized network environment. The first stage of game is the bandwidth negotiation game where the SP requests bandwidth from multiple InPs. Every InP decides whether to accept or to deny the request when the SP would cause link congestion. The second stage is the bandwidth provisioning game, where different SPs compete for the bandwidth capacity of a physical link managed by a single InP. In this model, the authors did not take into consideration how every SP computes the bandwidth capacity requested which will be addressed in this paper.
Zaheer et al. [16] proposed an open market offering a fair competition environment for automated service negotiation and contracting in a network virtualization environment through auctioning. The model is a two-stage Vickrey auction model that is adjustable to diverse InP pricing models. It offers to the SP a partitioning heuristic that minimizes the cost of VN setup. It not only considers intra-InP price, but also the preference of each SP for resource co-location and the high cost of inter-InP communication.

Allocating adequate bandwidth is a key for the InP to ensure the network performance. In this work, we present fine-grained and on-demand bandwidth allocation for multiple VNs. Each SP leases multiple physical links from different InPs to deploy its VNs. The SP specifies the per-hop latency for each VN. Then each VN has to compete to receive a fraction of the physical link's capacity. Our proposed mechanism computes the fraction that should be requested by the VN according to a QoS metric, such as the delay. We attempt to minimize the end to end delay for each VN while fairly and efficiently sharing the physical resources.

III. The system design

In this section, we present the system architecture model that will be used to perform a fair and efficient bandwidth allocation between multiple VNs. We propose a two-layer architecture similar to the one proposed by Padala et al. [17]. We adopt their work and extend it in order to fit the problem of dynamic bandwidth allocation in network virtualization. The goal of the proposed work is to prevent congestion collapse and to improve the fairness of bandwidth allocations. Each SP can deploy a single service on top of each VN. Then, it specifies target per-link delays. The proposed mechanism

computes the fraction request for each physical link to meet its delays requirements. When the sum of all the requests is greater than the link capacity, the mechanism uses a loss-load curve algorithm [18] to fairly distribute the link capacity between multiple VN and prevents from starvation. That provides a mathematical relationship between offered load and the level of packet loss at each physical link for each VN. It acts as a feedback mechanism for rate congestion control. In this work, we are interested in the dynamic resource allocation in order to reduce congestion of a physical link.

We model the physical network as an undirected graph and denote it by $G^v = \{N^v, L^v\}$, where N^v is the set of physical nodes and L^v is the set of the physical links. Each substrate link $l^v(i,j) \in L^v$ between two substrate nodes i and j is associated with the bandwidth capacity value C^l denoting the total amount of bandwidth. We suppose that the discovery step is already performed to find a set of potential InPs and their physical links to share. An optimized mapping of virtual nodes and virtual links on InP physical resources is also already achieved.

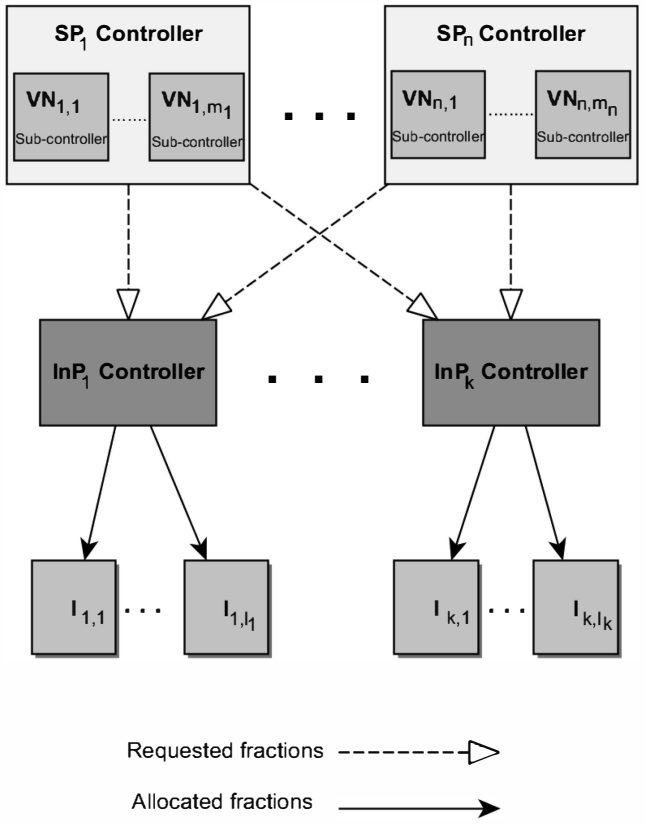

Fig. 1: The system architecture

Figure 1 explains the proposed architecture and how the controllers are interconnected. The model aims at minimizing the packet delays that occur when the packet arrival rate to the shared physical link exceeds that link's capacity. These controllers do not have any relationship with the OpenFlow controllers [19]. Since we are proposing a control-theoretic approach, we called each component of the proposed system "controller". The architecture includes a set of SP controllers and InP controllers. The SP controller is composed of a set of VN sub-controllers that monitor, for each VN, the dynamic workload changes at every control interval and compute the capacity fraction needed to achieve their performance.
The InP controller collects the capacity fraction requests from multiple SPs and determines, according to the loss-load curve algorithm, the fraction's allocation of each physical link. These controllers aim to allocate the maximum spare capacity to all the SPs, taking into consideration the capacity constraint of each managed link. For the sake of clarity, the notations used throughout this paper are given in Table I.

Symbol	Description
S	Set of all the active service providers SP_n
I	Set of all the infrastructure providers InP_k
V	Set of all the virtual networks $V_{i,j}$ deployed by a single service provider $s \in S$
L	Set of all the physical links $l_{k,l}$ managed by a single infrastructure provider $\text{InP}_k \in I$
$VN_{i,j}$	The virtual network with identifier j deployed by the service providers SP_i
T	Control interval
$x(T)$	Value of the variable x in control interval T
$Fr_{s,v,l}$	Fraction of the physical link $l \in L$ requested by the virtual network $v \in V$ that is deployed by the service provider $s \in S$
$Fa_{s,v,l}$	Fraction of the physical link $l \in L$ allocated to the virtual network $v \in V$ that is deployed by the service provider $s \in S$
$d_{s,v,l}$	The measured packet delay of the virtual network $v \in V$ deployed by $s \in S$ over the physical link $l \in L$
$\bar{d}_{s,v,l}$	The target packet delay of the virtual network $v \in V$ deployed by $s \in S$ over the physical link $l \in L$
$p_{s,v,l}$	Packet loss probability of the virtual network $v \in V$ using the link $l_{k,l}$
$y_{s,v,l} = 1/d_{s,v,l}(T)$	The measured performance of the virtual network $v \in V$ over the physical link $l \in L$
$\bar{y}_{s,v,l} = 1/\bar{d}_{s,v,l}(T)$	The target performance of the virtual network $v \in V$ over the physical link $l \in L$
$\hat{y}_{s,v,l} = y_{s,v,l}/\bar{y}_{s,v,l}$	The normalized delay of the virtual network $V_{i,j}$ over the physical link $l \in L$
$k(T)$	Behaviour parameter of the loss-load approach

TABLE I: Notations for the controller system

A. The SP controller

As stated above, the service provider attaches a sub-controller to each of its deployed VNs in order to find the optimal fraction requests that maintain at a fixed rate the packet

delays between two physical nodes according to the dynamic change in the workload. This sub-controller is responsible for monitoring past performance and allocation in order to find the optimal fraction request. The SP controllers collect all the requested fractions and submit a fraction request vector to each InP. The VN sub-controller has two modules: the estimator and the requester. The estimator updates the relationship between the allocated fractions and the performance of each VN. The requester is responsible for predicting the fraction request for each $VN_{i,j}$ to meet its target delay based on the estimation model. In this research, we assume that the behaviour of the VN can be locally approximated, at a single control interval T, by a linear model. This model captures the relationship between the allocations and the performances in terms of the packet delays of each $VN_{i,j}$ deployed by a single SP_j. Each SP submits to each InP a fraction request vector Fr_s. This vector represents all the requests for its deployed VNs on the different physical links managed by a single InP.

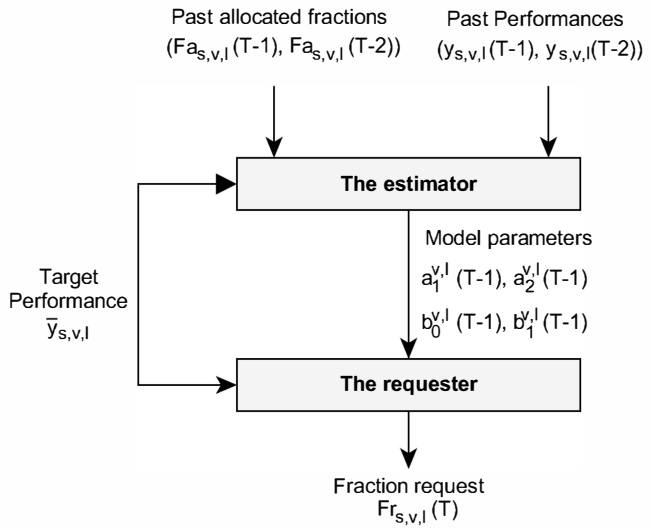

Fig. 2: The VN sub-controller architecture

1) The estimator: The estimator uses a linear adaptive model where the allocated fraction of each physical link and the VN packet delays over that link are related. The model parameters are updated, each control interval T, using the Recursive Least Squares (RLS) algorithm [20]. Periodically, the estimator learns and adaptively changes the parameters using the following equation :

$$\hat{y}_{s,v,l}(T) = a_1^{v,l}(T)\,\hat{y}_{s,v,l}(T-1) + a_2^{v,l}(T)\,\hat{y}_{s,v,l}(T-2) + b_0^{v,l}(T)\,Fa_{s,v,l}(T) + b_1^{v,l}(T)\,Fa_{s,v,l}(T-1) \quad (1)$$

The parameters $a_1^{v,l}(T)$ and $a_2^{v,l}(T)$ capture the correlation between the VN's past and actual performance over the physical link l. $b_0^{v,l}(T)$ and $b_1^{v,l}(T)$ are two vectors to capture the correlation between the past and the actual allocation over the link l. The adaptive models parameters depend on the control interval. Due to the RLS algorithm recursive nature, the computation time of these parameters is significantly reduced. We use the normalized performance rather than the absolute performance in order to ensure the stability of our system since the fraction requested and the fraction allocated have values between 0 and 1.

2) The requester: Once the requester receives the estimation of the linear model parameters, its role is to predict the fraction of each physical link that is required to meet the packet delay target over that link for $VN_{i,j}$ that is deployed by the service provider SP_j. The requester aims at finding the optimal fraction requests to meet its end to end delay targets. This is performed by finding the value of $Fr_{s,v,l}(T)$ that minimizes the following cost function :

$$Fr_{s,v,l}(T)^* = [\hat{y}_{s,v,l}(T) - a_1^{v,l}(T)\,\hat{y}_{s,v,l}(T-1) - a_2^{v,l}(T)\,\hat{y}_{s,v,l}(T-2) - b_1^{v,l}(T)\,Fa_{s,v,l}(T-1)]\,/\,b_0^{v,l}(T) \quad (2)$$

The parameters a_1, a_2, b_0, and b_1 have new updated values, at T control interval, using the RLS algorithm.

B. The InP controller

In this subsection, we present the fraction allocation algorithm adopted by the InP controller to allocate a portion of the available link capacity to multiple VNs. The InP controller receives the fraction requests from multiple SP controllers. Then, the controller sequentially collects the fraction requests of each VN from each physical link. The controller uses a fraction allocation algorithm to find the optimal allocation according to each link capacity, using the loss-load curves approach [19]. This approach captures the mathematical relationship between the load offered to each VN and the level of its packet loss. It offers a good feedback mechanism for sharing the available link capacity. The controller acts as a rate-based reactive congestion control scheme. It punishes greedy VNs by giving them less link capacity than they would get if they were less greedy.

Let us consider that there are M virtual networks deployed by N service providers and they are sharing a physical link l managed by a single infrastructure provider. The InP controller receives the fraction requests of all the VNs that are asking for a portion of the available link capacity.
The loss-load formula captures the probability of packet loss for the M^{th} VN according to the traffic load generated by the other $M-1^{th}$ virtual network. For each managed physical link l, the InP computes the total fraction requested by all the VNs that are sharing that link as:

$$Fr_l(T) = \sum_{s=1}^{N} \sum_{v=1}^{m_s} Fr_{s,v,l}(T) \quad (3)$$

If $Fr_l \leq 1$, then each VN receives a fraction allocation equal to its fraction request. Otherwise the controller computes two parameters, $\alpha(T)$ and $\beta(T)$ as:

$$\begin{cases} \alpha(T) = Fr_l(T) - 1 \\ \beta(T) = \sum_{s=1}^{N} \sum_{v=1}^{m_s} Fr_{s,v,l}^{k(T)+1} \end{cases} \quad (4)$$

The parameter $k(T)$ defines the behaviour of the loss-load approach at T instant. The parameter $\alpha(T)$ measures the excess traffic load at the link and $\beta(T)$ measures the distribution of the total load among the competing VNs.
The boundary conditions for the loss-load curve, $Fr_0(T)$ and $Fr_m(T)$ are defined as :

$$\begin{cases} Fr_0(T) = (\beta(T)/\alpha(T))^{1/k(T)} \\ Fr_m(T) = 1 - Fr_l(T) \end{cases} \quad (5)$$

The loss-load algorithm has the following properties for assigning the $p_{s,v,l}$:

- If the sum of all the requested fractions is less than or equal to the available capacity, then $p_{s,v,l} = 0$ for all the VNs.
- If the sum of all the requested fractions is greater than the physical link capacity, for all the concurrent VNs we have $p_{s,v,l} > 0 = Fr_{s,v,l}^{k}\,\alpha(T)/\beta(T)$.
- The $p_{s,v,l}$'s is a non-decreasing functions of the fraction requests.

The probability of packet loss $P(Fr(T))$ for the M^{th} virtual network at the control interval T when requesting a fraction $Fr(T)$ of the capacity C_l of a link L with $M-1$ other active VNs using the loss-load is given by:

$$\begin{cases} P(Fr(T)) = \dfrac{Fr^{k(T)}(Fr(T)+\alpha(T))}{Fr^{k+1}(T)+\beta(T)} \\ \qquad\qquad if\ Fr_0(T) \le Fr(T) \le Fr_m(T) \\ P(Fr(T)) = 1 \qquad if\ Fr(T) > Fr_m(T) \\ P(Fr(T)) = 0 \qquad if\ Fr(T) < Fr_0(T) \end{cases} \quad (6)$$

The loss-load formula is expressed as a function of all the received fraction requests and the current load parameters $\alpha(T)$ and $\beta(T)$ at the physical link. The term $Fr(T)+\alpha$ represents the new excess load at the link if a new fraction of the capacity is requested. The denominator represents the new measure of the total load distribution $Fr^{k(T)}(T)$ and determines the $(M+1)$st VN's share of the total packet loss required at the physical link.
Using a loss-load approach helps the controller to punish greedy VNs when they ask for a high fraction request. A physical link is considered critical if, at the control interval T, the sum of the bandwidth request of all the VNs is greater than the available capacity C_l.
For each link, $k(T)$ is set at every control interval T according to the link state in the previous control intervals. This parameter determines the behaviour of the loss-load algorithm to allocate the link's capacity to multiple VNs. It is given by the following equations:

$$\begin{cases} k(T) = 1\ if\ \ T = 0 \\ k(T) = 2 * k(T-1) \\ \qquad\qquad if\ Fr_l(T-1) > 1\ and\ Fr_l(T) > 1 \\ k(T) = k(T-1) - 1 \quad otherwise \end{cases} \quad (7)$$

Algorithm 1 Fraction allocation algorithm

Input: Virtual network v : $VN_{i,j} \in V$
Service provider $SP_j \in S$
Infrastructure provider $InP_m \in I$
Physical link l : $l_{m,n} \in L$
Control interval T
Total fraction requested $f_l(T-1)$
Bandwidth request $Fr_{s,v,l}(T)$
Loss-load behaviour parameter $k(T-1)$

Output: Fraction allocation $Fa_{s,v,l}(T)$

for each control interval T **do**
  for each physical link $L_{m,n}$ **do**
    for each service provider SP_j **do**
      for each virtual network $VN_{i,j}$ **do**
        $Fr_l(T) = \sum_{i=1}^{N} \sum_{j=1}^{M} Fr_{s,v,l}(T)$
        if $Fr_l(T) \le 1$ **then**
          $Fa_{s,v,l}(T) = Fr_{s,v,l}(T)$
        else
          if $Fr_l(T-1) > 1$ **then**
            $k(T) = 2 * k(T-1)$
          else
            $k(T) = k(T-1) - 1$
          end if
          $\alpha(T) = Fr_l(T) - 1$
          $\beta(T) = \sum_{s=1}^{N} \sum_{v=1}^{m_s} Fr_{s,v,l}^{k(T)+1}$
          $p_{s,v,l}(T) = Fr_{s,v,l}^{k(T)}\,\alpha(T)/\beta(T).$
          $Fa_{s,v,l}(T) = (1 - p_{s,v,l}(T)) * Fr_{s,v,l}(T)$
        end if
      end for
    end for
  end for
end for

The node controller uses a fraction allocation algorithm based on the loss-load approach to share the available link capacity among the concurrent VNs. In fact, this algorithm takes into account the state of the physical links at the previous control intervals and the bandwidth requests of all the VNs. Then, it attempts to find the optimal bandwidth allocation. Algorithm 1 presents the different steps followed by the node controller to allocate bandwidths to multiple VNs.

IV. Experiments and results

In this section, we present performance evaluations of the proposed allocation scheme that are done by MATLAB simulations. To reduce the complexity of the simulation, we supposed that three VNs, $\{VN_{1,1}, VN_{2,1}, VN_{3,1}\}$ deployed by three different SPs, are sharing a single physical link

$l_{1,1}$ managed by an InP_1. We conducted experiments with OPNET to capture a trace file containing the allocation and delay for three different services that are sharing the same physical link. The control interval is equal to T=1s. Then, the fraction request algorithm is executed offline to find the optimal fraction request for each VN. The target delays of each VNs are maintained constant at each control interval. When the sum of all the requests exceeds the link capacity, we analyze the fraction allocation of our proposed algorithm, named fraction allocation algorithm (FAA), and compare our results with proportional share algorithm (PSA). We investigate the fairness and efficiency of the obtained results.

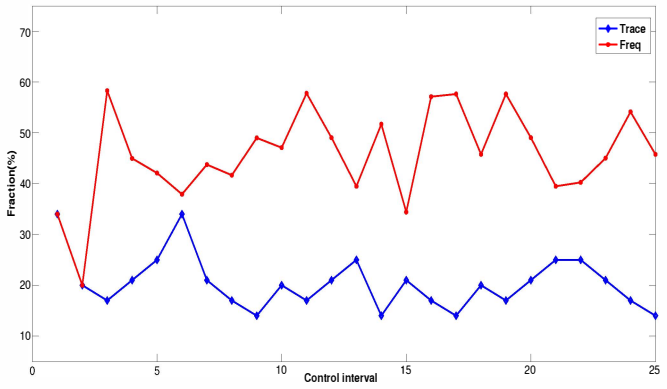

Fig. 3: The fraction of the physical link $l_{1,1}$ requested by the virtual network VN_1 at each control interval

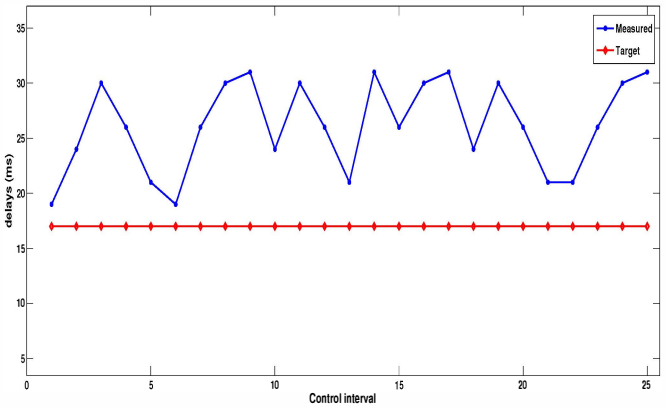

Fig. 4: The measured and the target packet delays for virtual network VN_1 at each control interval

Figure 3 shows the fraction requested by the virtual network $VN_{1,1}$ to receive a portion of the physical link $l_{1,1}$. Figure 4 depicts the measured and the target delays over the shared link. As shown in both figures, our algorithm reacts adequately to workload changes over the physical link. Each VN monitors its past performance in terms of packet delay and its allocation in terms of bandwidth. Then it captures the relationship between them to find its optimal fraction request over the physical link. The goal is to maintain the packet delay at 17 ms at each control interval.

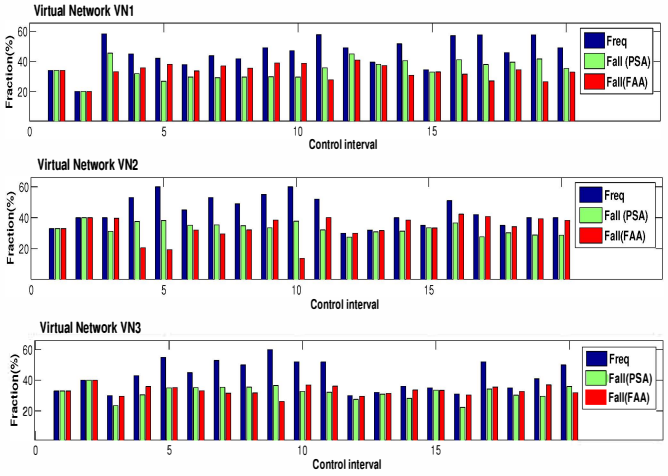

Fig. 5: The fraction allocated of the physical link $l_{1,1}$ to the virtual networks VN_1, VN_2, and VN_3

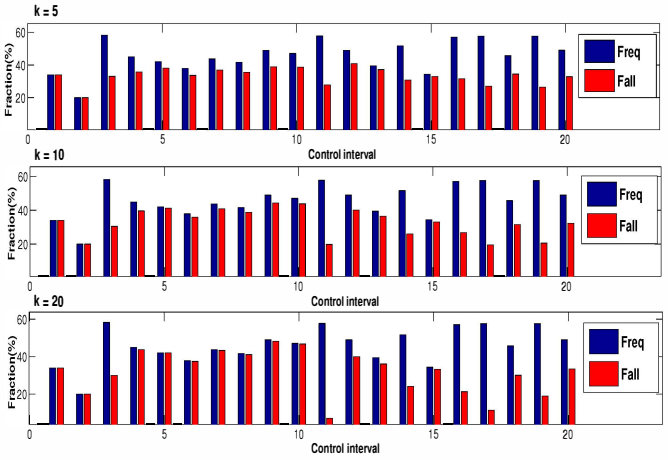

Fig. 6: The impact of the loss-load behaviour parameter k in the fraction allocation

Figure 5 illustrates the fraction requested of the physical link $l_{1,1}$ by the three VNs and the allocated fraction using the PSA algorithm and our proposed allocation algorithm FAA. At each control interval, we compare the allocated fraction by both algorithms to each VN when the behaviour parameter of the loss-load algorithm is fixed to k=5. It is clear that our algorithm offer better results than the proportional share one when there is a greedy VN that is asking for a greater fraction than the two other VNs. For instance, at T=10, the virtual network $VN_{2,1}$ asks for a fraction Fr=62%. It is considered a greedy VN because $VN_{1,1}$ asked for Fr=47 and $VN_{3,1}$ asked for Fr=49. Our algorithm satisfies the request of the non-greedy VNs and punishes the greedy ones by giving them less fraction than they would get if they were less greedy.

Figure 6 depicts the fraction requested and the fraction allocated of the link $l_{1,1}$ to the virtual network $VN_{1,1}$. Through the simulation, we are analyzing the impact of the loss-load behaviour parameter k on the fraction allocation. The goal is to discuss the degree of punishment of the greedy VN, which depends on the loss-load behaviour parameter k. For example, at T=11, the virtual network $VN_{1,1}$ is considered greedy. When k=10, it receives 28% of the physical link capacity. When the parameter k is equal to 20, which means the physical link $l_{1,1}$ as been critical for a long time, the greedy VN receives only 11% of the link capacity. Our algorithm prevents starvation by satisfying the request of the non-greedy VNs and punishing the greedy ones.

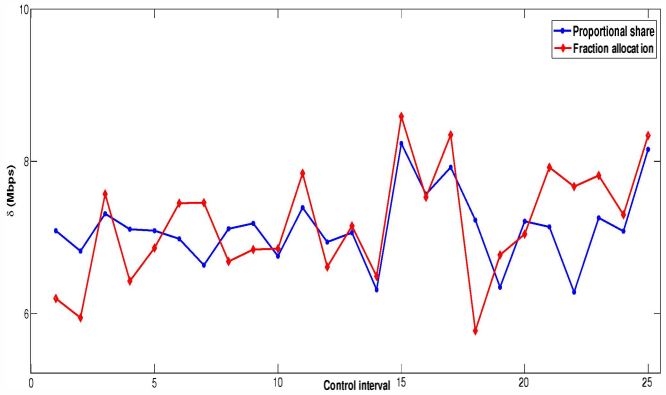

Fig. 7: The mean difference between the requested and the allocated fractions using PSA and FAA

In order to analyze the fairness of our proposed allocation scheme, we ran our algorithm 250 times with different fraction requests. We computed δ, which is the mean difference between the fractions requested and the fractions allocated to the three VNs, during 10 control intervals and we compare it to the one obtained using PSA algorithm. Figure 7 illustrates the obtained results. In fact, the difference is almost the same for both algorithms. Since the proportional share algorithm is shown to provide fair bandwidth shares, we may conclude that our algorithm offers a fair allocation mechanism for multiple VNs sharing the same physical links.

V. Conclusion

In this paper, we have presented a two-layer controller system for dynamic bandwidth allocation in a virtualized network environment, where each SP leases link capacity from multiple InPs. The proposed system is composed of SP controllers and InP controllers. The SP controller is in turn composed of a set of VN sub-controllers that are responsible of estimating and optimizing the VN fraction requests of the physical links at every control interval. The InP controller is responsible for allocating the available link capacity between multiple VNs deployed by different SPs. The objective of the proposed work is to offer an autonomous bandwidth allocation for multiple VNs. We also aim to provide a fair and efficient allocation of link capacity and avoid bottlenecks. In the future, we need to implement our work in a real network environment, in order to analyze its applicability and the limitations of our model. Also, the mapping of VNs to specific nodes and links in the substrate network should be taken as a constraint, in order to lead to better results for bandwidth allocation in network virtualization.

References

[1] N.M.M.K. Chowdhury and R. Boutaba, "A survey of network virtualization", *Journal of Computer Networks*, vol.54, no.5, pp. 862-876, 2010.

[2] N.M.M.K. Chowdhury and R. Boutaba, "Network virtualization: state of the art and research challenges," *IEEE Communications Magazine*, vol.47, no.7, pp.20-26, 2009.

[3] A. Haider, R. Potter and A. Nakao, "Challenges in resource allocation in network virtualization", In *Proc. of 20th ITC Specialist Seminar*, 2009.

[4] F. Esposito, I. Matta, and V. Ishakian, "Slice Embedding Solutions for Distributed Service Architectures", *The ACM Computing Surveys Journal*, 2012.

[5] M. Harchol-Balter, T. Leighton, and D Lewin, "Resource discovery in distributed networks", In Proc. of *The eighteenth annual ACM Symposium on Principles of distributed computing (PODC'99)*, pp. 229-237, 1999.

[6] W. Hsu, Y. Shieh, "Virtual network mapping algorithm in the cloud infrastructure", *Journal of Network and Computer Applications*, 2013.

[7] A. Fischer, J. F. Botero, M. Beck, H. De Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *IEEE Communications Surveys and Tutorials* , vol.PP, no.99, pp.1-19, 2013.

[8] M.A. Wang, R. Dutta, G. N. Rouskas, and I. Baldine, "Network Virtualization: Technologies, Perspectives, and Frontiers," *Journal of Lightwave Technology*, vol.31, no.4, pp.523,537, 2013.

[9] A. Belbekkouche, M. M. Hasan, A. Karmouch, "Resource Discovery and Allocation in Network Virtualization", *IEEE Communications Surveys and Tutorials*, vol.pp, no.99, 2012, pp. 1-15.

[10] J.F. Botero, X. Hesselbach, "The bottlenecked virtual network problem in bandwidth allocation for network virtualization," In *Proc. of EEE Latin-American Conference on Communications*, pp.1-5, 2009.

[11] M.R. Rahman, I. Aib, and R. Boutaba, "Survivable virtual network embedding", In *Proc. of the 9th IFIP TC 6 International Conference on Networking*,pp. 40-52, 2010.

[12] J. He, R. Zhang-Shen, Y. Li, C. Lee, J. Rexford, and M. Chiang. 2008. DaVinci: dynamically adaptive virtual networks for a customized internet. In *Proc. of 2008 ACM CoNEXT (CoNEXT '08)*, pp. 1-12, 2008.

[13] Y. Zhou, Y. Li, G. Sun; D. Jin, Li Su, and L. Zeng, "Game theory based bandwidth allocation scheme for network virtualization," In *Proc. of the IEEE Global Telecommunications (GLOBECOM 2010)*, pp.1-5, 2010.

[14] C. Wang, C. Wang, and Y. Yuan, "Game based dynamical bandwidth allocation model for virtual networks," In *Proc. of the 1st International Conference on Information Science and Engineering (ICISE)*, pp.1745,1747, 2009.

[15] M.S. Seddiki, M. Frikha, "A non-cooperative game theory model for bandwidth allocation in network virtualization," In *Proc. of XVth International Telecommunications Network Strategy and Planning Symposium*, pp.1,6, 2012.

[16] F.E. Zaheer, X. Jin, and R. Boutaba, "Multi-provider service negotiation and contracting in network virtualization," In *Proc. of Network Operations and Management Symposium (NOMS)*", pp.471-478, 2010.

[17] P. Padala, K. Hou, K. G. Shin, X. Zhu, M. Uysal, Z. Wang, S. Singhal, and A. Merchant. "Automated control of multiple virtualized resources", In *Proc. of the 4th ACM European conference on Computer systems*, pp.13-26, 2009.

[18] C.L. Williamson, "Dynamic bandwidth allocation using loss-load curves," *IEEE/ACM Transactions on Networking*, vol.4, no.6, pp.829,839, 1996.

[19] 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, pp.6974, 2008.

[20] Y. Engel, S. Mannor, R. Meir, "The kernel recursive least-squares algorithm," *IIEEE Transactions on Signal Processing* , vol.52, no.8, pp.2275-2285, 2004.