A New Mechanism for SDN Network Virtualization Service

Changcheng Huang
Department of Systems and Computer Engineering
Carleton University
Ottawa, Canada
huang@sce.carleton.ca

Jiafeng Zhu, Min Luo, Wu Chou
Huawei Technologies Inc.
Santa Clara, US
{jiafeng.zhu, min.ch.luo, wu.chou}@huawei.com

***Abstract*— Software Defined Network (SDN) is becoming a new paradigm for both WAN and enterprise networks. By separating control and data planes, SDN allows network providers to sell new services without changing their physical switches. One of the key services SDN can enable is network virtualization which allows a service provider to have a virtual slice of a network provider's physical network. While this feature empowers service providers to have dynamic networks, it is becoming more difficult for a network provider to statistically share its resource usage across multiple service providers due to the fact that service providers are reluctant to share their user information. This paper presents a new scheme that empowers network provider to maximize statistical resource sharing with service guarantee while minimizing information shared.**

Keywords—Software Defined Network, Virtual Topology, OpenFlow, OFC

I. INTRODUCTION

Research on SDN started nearly ten years ago. The motivation at that time was to design a new network architecture that allows network managers to have more flexible control of their networks. Ethane [1], for example, was proposed in 2007 for enterprise networks. Ethane deployed simple flow-based Ethernet switches with a centralized controller that manages the admittance and routing of flows. Ethane was implemented both in hardware and software with more than 300 wired or wireless hosts.

OpenFlow [2] was proposed in 2008. Initially it was proposed for researchers to run experimental protocols in the networks they use every day. It turned out OpenFlow has multiple benefits. On one hand, it allows researchers to run experiments on heterogeneous switches in a uniform way at line-rate and with high port density; on the other hand, vendors do not need to expose the internal workings of their switches. Due to this feature, OpenFlow has gained traction in vendor and network provider communities in an unexpected speed. In 2011, Open Networking Foundation (ONF) [3] was formed. Member companies now include many network equipment vendors, semiconductor companies, computer companies, software companies, startups, telecom service providers, hyper-scale datacenter operators, and enterprise users. By 2012, member companies had introduced 64 OpenFlow products to market and over 30 million OpenFlow-capable ports had shipped.

Most of earlier SDN networks were designed for datacenter infrastructure [4] where flexibility and scalability are critical. With tens of thousands commodity switches and servers, datacenter presents a serious challenge to network management. SDN solves the problem by separating control plan from data plane. With relatively an independent control plane, SDN does not need to be run on proprietary equipment. Instead, the control plane can be implemented with a large number of regular servers. By using a centralized structure, these servers can be efficiently supported by datacenter with on-demand service capacity.

In recent years, SDN has found applications in other areas. Google, for example, built a large scale SDN-based WAN network [5] that has attracted attentions worldwide. Large carriers are looking at the possibility of upgrading their network infrastructures with SDN architecture. One issue to be solved first is scalability. Some recent study has investigated the scalability issue related to SDN architecture. Early benchmarks on NOX [6] (the first SDN controller) showed it could only handle 30,000 flow initiations per second while maintaining a sub-10 ms flow install time. Recent works have shown that SDN scalability can be extended by using multicore systems [7] or deploying multiple OpenFlow controllers (OFCs) [8-9].

With a separated control plane, SDN has the potential to enable new services. One of the primary new services that have been envisioned is the virtual topology service, which allows network provider to sell different virtual topologies to different service providers. Each service provider can use its virtual topology just like the way it uses its own private network while sharing underlying physical network resources with other service providers. The network provider, on the other hand, can enjoy new revenue growth through selling virtual topologies with different granularities. This benefit, however, does not come without a challenge. One of the key issues is the division of ownership, which makes statistical resource sharing more difficult. This issue has not been fully addressed yet. This paper is targeted at this issue in specific. Our goal is to provide statistical guarantee to service providers while allowing network provider to enjoy multiplexing gain.

The paper is organized as follows. In Section II, we will discuss more detail about the issue to be resolved. In Section III, we will propose a new and practical scheme that allows

each entity to exercise its own authority while achieving gain in resource saving. This will be followed by more description about how parameters will be calculated in Section IV. Numerical results will be presented in Section V. We will finish the paper with some concluding remarks in Section VI.

II. TOPOLOGY ABSTRACTION AS A SERVICE

The initial SDN adopters, both vendors and network providers, have focused on some fundamental issues related to separating control and data planes. The benefits of network virtualization have not been fully explored. A good example is the Google SDN WAN project mentioned earlier [5]. Before the SDN project, the WAN Google had been using for interconnecting their datacenters had been managed using traditional approach which was slow due to manual provisioning process. On average, the utilization of Google's WAN at that time was 20 to 30%. Through multiple years of efforts, Google has upgraded its WAN with SDN technology. The initial results are very encouraging. Utilization has been increased to around 70 to 90%. In some cases, 100% utilization has been achieved. The key enablers of this improvement are the SDN dynamic flow creation capability and a more sophisticated optimization approach based on Max-min fairness. SDN allows Google to do centralized traffic engineering that can balance load distribution across their entire WAN with the sophisticated algorithm. Large amount of elastic traffic also helps increase the utilization to 100%.

However, it should be noted that Google's WAN is a special case where Google is the user, service provider, as well as network provider, i.e., Google provides service to itself on its own network. This nature allows Google to do global optimization easily. For example, because Google can control the traffic carried by the WAN, operators can decide when and how long the elastic traffic will be buffered. Also because Google owns both service network and underlying WAN network, operators can have a global view of the network and therefore optimize network usage globally.

In a real world, it is quite common that users, service providers, and network providers are separate entities. They may all have their own objectives which may conflict with each other. Take the example of enterprise network. With the fast growth of datacenters, enterprises are becoming increasingly interested in outsourcing their enterprise networks to datacenters. Under this scenario, the owner of the network now is the owner of the datacenter, such as Amazon. The service providers are the enterprises who outsource their enterprise networks to the datacenters. Therefore a service provider is independent from the network provider as well as independent from other service providers who share the same physical network. Recognizing this need, the pioneers of SDN advocate a layer called FlowVisor [10] which plays the same role as the hypervisor for virtual machines. FlowVisor allows a network provider to partition its physical network into slices for various service providers. Each service provider can virtually own one slice which has its own virtual network topology and related resources generated through a topology abstraction process. The service provider can then optimize its usage of the slice which it owns. A network provider can sell virtual topology service to service providers with different granularities selling at different prices [11]. This will change the situation that network providers today can only sell pipes and equip network providers a new venue for revenue growth. The SDN architecture for topology abstraction service is shown in Figure 1. As shown in the figure, each entity has its own OpenFlow Controller (OFC). The OFC of a service provider is in charge of routing user flows and optimizing resource usage within its own virtual topology. The OFC of the network provider will execute the topology abstraction process through a topology virtualizer based on a global topology map of the underlying physical network.

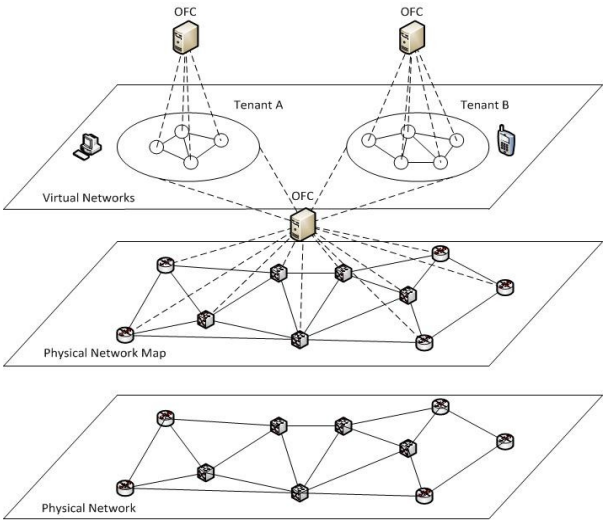

Fig. 1. SDN architecture for virtual network service.

While virtual topology allows a network provider to sell topology abstraction service to service providers, it also makes the network provider partly lose the control of its resource allocation capability. After selling a virtual topology to a service provider, the network provider has no control of how user traffic will be routed within the virtual topology owned now by the service provider. This makes the network provider difficult to optimize its resource usage across multiple service providers. There are several situations that make the issue particularly thorny:

- A service provider may not be legally allowed to disclose information about the traffic of its individual users (also called micro-flow information) to the network provider although service providers can provide aggregate information such as mean and peak rates of a virtual link. A virtual link typically carries large number of micro-flows.
- A service provider may be a competitor of the network provider. For example, RIM uses Amazon cloud service while it is also a competitor with Amazon in tablet devices.
- The network provider may not want to disclose its network usage information to a service provider in afraid of the service provider using the information to bargain.

In general, each entity tends to disclose as little information as possible due to various legal and commercial reasons. Therefore the topology abstraction process is also an information-filtering process that tries to minimize the information exchanged between different entities.

The reality that there are multiple authorities with limited shared information has made resource sharing extremely difficult. This issue has not been addressed in literature as well as in practice. Existing mapping solutions [12-13] for virtual topology services are typically based on the assumption that service providers will provide a bandwidth requirement for each virtual link to their network provider. The network provider then generates the virtual topologies with required bandwidths. No statistical sharing among the reserved bandwidths for different virtual links sharing a physical link is allowed. On the other hand, it is also difficult for a service provider to know exactly how much traffic it can put on a virtual link without suffering performance degradation because the service provider has no information about the characteristics of other virtual networks sharing the same physical network. This leaves a service provider no choice but to conservatively engineer the peak rate on each virtual link to be below the guaranteed fixed bandwidth. While this makes the formulation of the virtual network mapping problem easier, it is a loss of network utilization for the network provider because it cannot take advantage of the dynamic nature of the traffic carried by each virtual link. In this paper, we call this kind of service as wired virtual topology service.

In the following section, we will propose a new service and a practical and effective mechanism that will help a network provider maximize its network utilization while providing statistical guarantee to service providers.

III. VIRTUAL TOPOLOGY WITH STATISTICAL GUARANTEE

We consider the case that network provider and service providers are separate entities. Service providers buy their virtual topologies from a network provider. Different virtual topologies share a physical network owned by the network provider. The OFC of each entity is trying to maximize its own profit through maximizing its virtual/physical network utilization. We propose a new service called elastic virtual topology service that allows a network provider to utilize its resources more efficient while minimizing the information to be shared between the network provider and its clients, service providers.

With the elastic virtual topology service, a service provider receives a statistical guarantee that promises a) the service provider can send its traffic on each virtual link as it is with a negotiated low congestion probability; b) the service provider can still have a negotiated minimum bandwidth when congestion happens. In return, the service provider needs to provide some basic characteristics of its traffic carried by each virtual link. It should be noted that each virtual link may carry numerous micro-flows. The information to be shared is the aggregated traffic characteristics of the virtual link rather than individual micro-flows. In this way, the service provider can still hide information about its users from network provider while the latter can achieve scalability through handling aggregate traffic instead of individual micro flows.

In specific, we propose the following Virtual Topology with Statistical Guarantee (VTSG) scheme. To start with, a service provider sends a request to network provider with a description of the requested virtual topology and associated mean and peak rate for each virtual link instead of sending topology request with associated bandwidths. Different from bandwidths, the mean and peak rates of a virtual link are decided by the user traffic and routing algorithm used by the service provider, which are under the control of the service provider. Therefore it is possible for the service provider to estimate these statistics based on its user profile. On the other hand, the bandwidth received by a virtual link will be dynamic and depend on traffic characteristics of other virtual links sharing the same physical link for statistical multiplexing gain and can only be decided by the network provider who has information about traffic characteristics of all virtual links sharing the physical link.

The network provider will use the mean and peak rates to calculate a minimum bandwidth that is higher than the mean rate of the virtual link but smaller than its peak rate with a high percentage (say 99.9%) guarantee that the virtual link will be able to send its traffic as it is without congestion. When congestion does happen, the virtual link will be guaranteed with the minimum bandwidth. When a virtual link traverses multiple physical links, the minimum bandwidth the virtual link can get will be the smallest of the minimum bandwidths supported by all the involved physical links.

Equipped with a way to calculate the minimum bandwidth required for each virtual link, the network provider will then map the virtual topology by formulating it as an optimization problem with the minimum bandwidths as the demands. To this end, there are various existing approaches such as those in [12-14] can be applied. So our focus in this paper will be on the calculation of the minimum bandwidth rather than on the formulation of the mapping problem. It is important to note that the complexity of calculating the minimum bandwidth has a significant impact on the complexity of the mapping algorithms. That is why existing approaches [12-14] assume the bandwidth requirement for each virtual link is simply a constant which does not depend on other virtual links sharing the same physical link. Our goal is to find a way that existing approaches such as those in [12-14] can be directly applied without any changes.

After mapping is done, the network provider will then notify the service provider of the minimum bandwidth for each virtual link and the guaranteed performance. Without congestion, service providers are allowed to send their natural traffic as it is. When congestion happens at a physical link, the network provider will send alerts to the service providers sharing the physical link and shape those virtual links with elastic virtual topology service to their minimum bandwidths.

Upon receiving alerts, those service providers will decide how to route their user traffic within their virtual networks to avoid their congested virtual links and minimize the impact on their virtual networks. For example, if a service provider is using OSPF routing protocol, the service provider can simply increase the costs of those congested links. Traffic will then be diverted to other uncongested links. Certainly, other more

sophisticated optimization approaches can be used with SDN centralized control architecture. It is important to note that our VTSG approach allows network provider and service provider to make their separate resource usage decisions while sharing minimum amount of information.

Our objective in the next section is to develop a method for calculating the minimum bandwidth for each virtual link so that a) it allows statistical multiplexing across different virtual links sharing the same physical link; b) all existing topology mapping algorithms such as those in [12-14] can still be applied using the minimum bandwidths as virtual link demands.

IV. MINIMUM BANDWIDTH CALCULATION

Under the Quality of Service research area, numerous ways have been developed to calculate congestion probability and bandwidth requirement (e.g. [15]). Most of these approaches require detail traffic characteristics of underlined micro-flows and complex analytical manipulations. Under our context, micro-flow information of individual users is not available and complex analytical manipulations will make the topology mapping becoming more complex and intractable. Our objective is to find a practical approach that allows a SDN controller of a network provider to calculate the minimum bandwidth for each virtual link under the VTSG service with limited information while achieving multiplexing gain. To meet these challenges, we propose the following approach.

We consider the scenario that a physical network has received *R* virtual network requests. The virtual links of the *R* virtual network requests constitute a set *L*. We assume that each service provider will inform the network provider of its requested virtual network topology and mean and peak rates of the traffic associated with each virtual link. The information shared is aggregate because each virtual link carries large number of micro-flows. This kind of information sharing is the minimum required for virtual topology service.

A simple example is shown in Figure 2 where a physical network includes four nodes (Nodes A, B, C, D) and five links (Links (A, B), (B, C), (C, D), (D, A) (A, C) as shown in solid lines). Two virtual topologies are generated out of the physical network. One virtual topology includes virtual nodes A, B, C and virtual links $(A, B)_1$, $(B, C)_1$, $(A, C)_1$ as shown in dashed lines. The other includes virtual nodes A, C, D and virtual links $(A, C)_2$, $(C, D)_2$ and $(D, A)_2$ as shown in dash-dotted lines. Here we assume virtual nodes happen to overlap with physical nodes to simplify our notations. It is easy to see that physical link (A, C) carries two virtual links $(A, C)_1$ and $(A, C)_2$.

Due to the fact that we do not have information about the characteristics of individual micro-flows within a virtual link, we need to capture the nature of the aggregate traffic with a virtual link directly. To this end, we assume that the traffic carried by each virtual link can be modeled by a Gaussian process. This is a valid assumption in most cases because a virtual link typically carries a large number of micro-flows which lead to Gaussian process based on Central Limit Theorem. Many measurements have confirmed that aggregate Internet traffic indeed follows Gaussian processes [16].

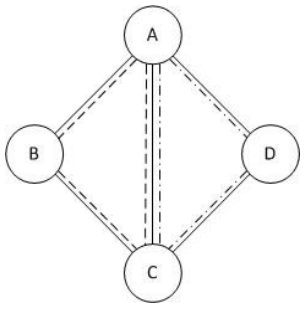

Fig. 2. A sample physical network with two virtual topologies generated.

In general, we assume each virtual link l, where $l \in L$, carries Gaussian traffic with mean rate m_l and peak rate p_l as announced by the corresponding tenant. After receiving mean rate m_l and peak rate p_l for all $l \in L$, the network provider needs to calculate a minimum bandwidth b_l for each virtual link.

We denote the standard deviation of the Gaussian traffic with each virtual link as σ_l. We set q_l as the α-quantile of the Gaussian distribution where α_l will be decided by the network provider and can be kept constant for all virtual links. When α is very small, the quantile value q_l will happen rarely. Therefore q_l can be roughly treated as the peak rate, i.e.

$$p_l = q_l = m_l + \sigma_l \sqrt{2} erf^{-1}(1 - 2\alpha) \quad (1)$$

where *erf* ( ) is the error function of Gaussian distribution. The actual value for α will be decided by the service provider. We denote

$$T_\alpha = erf^{-1}(1 - 2\alpha)$$

This leads us to

$$\sigma_l = \frac{p_l - m_l}{\sqrt{2}\mathrm{T}_\alpha} \quad (2)$$

We first focus on the case that a virtual link will be mapped to a single physical link. We will discuss more general cases in the latter part of this section.

Suppose there are L_s virtual links sharing a substrate physical link *s*. Without loss of generality, we assume these virtual links are indexed 1, 2,…, L_s. It is easy to see that the aggregate traffic on the physical link is still a Gaussian process with a mean $m = \sum_{l=1}^{L_s} m_l$ and standard deviation $\sigma = \sqrt{\sum_{l=1}^{L_s} \sigma_l^2}$.

Similarly let q be the β-quantile of the aggregate traffic. Then we have

$$q = m + \sigma\sqrt{2} erf^{-1}(1 - 2\beta)$$
$$= m + \sigma\sqrt{2} T_\beta \quad (3)$$

If we provision the bandwidth capacity of the physical link as $b \geq q$, congestion will happen with a probability less than β. Therefore by controlling b, we can achieve any guaranteed congestion probability β.

The next issue is how to assign this capacity to individual virtual links so that a virtual link can have a minimum bandwidth guarantee when congestion does happen. We adopt a simple approach where the minimum bandwidth assigned to each virtual link is calculated as the following:

$$b_l \geq m_l + \sqrt{2}\frac{\sigma}{L_s}T_\beta \quad (4)$$

If Eq. (4) is satisfied, we will have

$$b = \sum_{l=1}^{L_s} b_l \geq \sum_{l=1}^{L_s}(m_l + \sqrt{2}\frac{\sigma}{L_s}T_\beta) = m + \sigma\sqrt{2}T_\beta = q$$

Therefore $b \geq q$ is satisfied. The reason for this simple approach is that the physical network has very little knowledge of the individual traffic flow routed in each virtual topology. This makes it very hard to apply other fairness approaches such as Max-min [5]. However while being simple as it is, it is still more complex than a constant bandwidth as existing algorithms have typically assumed.

Combining Eqs. (1) to (4), we can see that b_l depends on m_l's and p_l's of all the virtual links sharing the physical link as well as L_s, and furthermore the relationship is nonlinear. This nonlinear relationship will make all existing topology mapping algorithms such as those in [12-14] not applicable. The dependency on L_s will also make it infeasible because a network provider can only know L_s after optimization and mapping have been done, i.e., L_s is the mapping result rather than mapping input. Therefore we need to further simplify Eq. (4) so that it forms a linear relationship and it does not depend on L_s.

We consider the scenarios where the traffic carried by each virtual link is relatively small compared to the capacity of the physical link. In specific, we assume that the variance of the traffic within each virtual link is upper bounded, i.e., $\sigma_l \leq \Sigma$ for $l \in L$, where Σ is the upper bound. When very few virtual links are mapped to a physical link, the physical link is typically underutilized. The physical link will not get congested no matter how much minimum bandwidth is committed to each virtual link. So we will focus on those physical links hosting many virtual links, i.e. we assume the number of virtual links mapped to a physical link is lower bounded, which means $L_s \geq \Lambda$ for all physical links, where Λ is the lower bound. Then we have

$$m_l + \sqrt{2}\frac{\sigma}{L_s}T_\beta \leq m_l + \sqrt{2}\frac{\Sigma\sqrt{L_s}}{L_s}T_\beta \leq m_l + \sqrt{2}\frac{\Sigma}{\sqrt{\Lambda}}T_\beta$$

If we set

$$b_l = m_l + \sqrt{2}\frac{\Sigma}{\sqrt{\Lambda}}T_\beta \quad (5)$$

Eq. (4) will be satisfied. This means that the congestion probability for the virtual link l will be smaller than β. Although Eq. (5) is looser than Eq. (4), we can see that, when Λ goes to infinity, the allocated bandwidth for each virtual link with Eq. (5) converges to its mean rate which is clearly the minimum bandwidth any approach can assign with guaranteed performance. This is in clear contrast to peak rate allocation with the wired virtual topology service.

It is important to note that Eq. (5) is different from Eq. (4) in the sense that b_l is only dependent on m_l while all other parameters will be constant across all virtual links and all physical links. And furthermore, it does not depend on L_s. With this result, all existing topology mapping algorithms will be valid without any changes. So our objective has been achieved.

Now we will investigate the performance of our proposed VTSG scheme in comparison with the wired virtual topology service. Because the mean rate m_l and peak rate p_l vary from virtual link to virtual link, in order to study the average performance, we consider m_l's and p_l's as i.i.d. random variables respectively with $\bar{m} = E\{m_l\}$, $\bar{p} = E\{p_l\}$, $\bar{\sigma} = E\{\sigma\}$. Because virtual networks are typically independent, this assumption is roughly true in practice.

From Eq. (1), we have

$$\bar{p} = \bar{m} + \bar{\sigma}\sqrt{2}T_\alpha \quad (6)$$

From Eq. (5) we have

$$E\{b_l\} = \bar{m} + \sqrt{2}\frac{\Sigma}{\sqrt{\Lambda}}T_\beta \quad (7)$$

With the wired virtual topology service, the bandwidth required by each virtual link denoted by b_l^W will be its peak rate. We have

$$E\{b_l^W\} = \bar{p} \quad (8)$$

Now we can calculate how much bandwidth we can save in terms of percentage denoted by δ as

$$\delta = \frac{E\{b_l^W\} - E\{b_l\}}{E\{b_l^W\}} = 1 - \frac{\bar{m} + \sqrt{2}\Sigma T_\beta/\sqrt{\Lambda}}{\bar{p}} \quad (9)$$

From Eq. (9), we can see that, when Λ increases, i.e., the minimum number of virtual links sharing a physical link increases, the gain will also increase.

When the number of virtual links sharing a physical link is very small, the physical link becomes extremely unlikely to get congested. So it does not matter anymore in terms of how much bandwidth is assigned to each virtual link. Therefore, Eq. (5) can still be used.

When a virtual link is mapped to a path that traverses multiple physical links, we can calculate the minimum bandwidth required for each physical link using similar approach and then take their minimum as the minimum bandwidth for the virtual link. We will show some numerical examples in the next section.

If the traffic of a virtual link is splittable, we can model each portion of the splitted virtual link as a proportional amount of the whole virtual link traffic. This is typically the case if ECMP protocol is applied. The above approach can then be applied to each portion.

V. NUMERICAL RESULTS

We now examine some numerical results. We first want to see how the minimum bandwidth for each virtual link changes

with increasing number of virtual links sharing the same physical link. We assume each virtual link generates 1Gbps traffic in average and 1.3Gbps at peak. We set $\beta = \alpha$=0.1%. This means that the congestion probability for each physical link is set as low as 0.1%. The result is shown in Figure 3. It is easy to see that the minimum bandwidth goes down with increasing number of virtual links. More specifically, when there is only one virtual link using the physical link, the minimum bandwidth is its peak rate, i.e., no multiplexing gain. However this is typically the case that the physical link is underutilized due to the fact that the number of virtual links is too small. Therefore no congestion issue exists for this case.

When the number of virtual links starts increasing, the minimum bandwidth drops extremely fast at the beginning and flats out later. This shows that it is quite effective to multiplex a small number of virtual links.

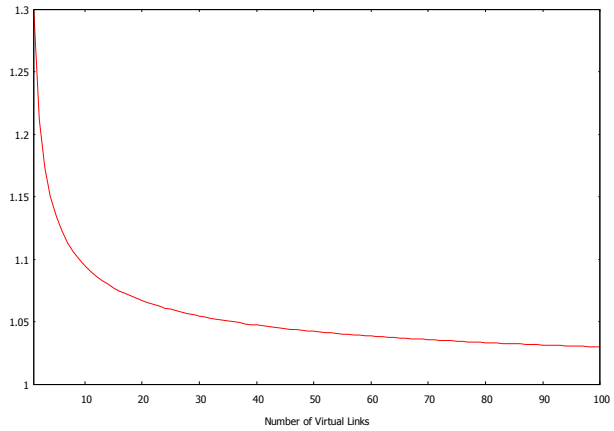

Fig. 3. Average minimum bandwidth for each virtual link vs. minimum number of virtual links sharing the same physical link. All virtual links have similar statistical characteristics. Each virtual link carries traffic with average mean rate of 1Gbps with average peak rate at 1.3 Gbps.

We now compare the two types of services. In order to simplify the parameter setting, we assume $\alpha = \beta$ and $\bar{\sigma} = \Sigma$, both are roughly true in real situation. We rewrite Eq. (8) as follows:

$$\delta = 1 - \frac{\bar{m} + \frac{\sqrt{2}\Sigma T_\beta}{\sqrt{\Lambda}}}{\bar{p}} = 1 - \frac{\bar{m} + \frac{\sqrt{2}\bar{\sigma} T_\beta}{\sqrt{\Lambda}}}{\bar{p}}$$

$$= 1 - \frac{\bar{m} + \frac{\sqrt{2}T_\beta \frac{\bar{p} - \bar{m}}{\sqrt{2}T_\alpha}}{\sqrt{\Lambda}}}{\bar{p}} = 1 - \frac{\left(1 - \frac{1}{\sqrt{\Lambda}}\right)\bar{m} + \frac{\bar{p}}{\sqrt{\Lambda}}}{\bar{p}}$$

$$= \left(1 - \frac{1}{\sqrt{\Lambda}}\right)\left(1 - \frac{\bar{m}}{\bar{p}}\right)$$

We define $v = \frac{\bar{\sigma}}{\bar{m}}$ as the coefficient of variation for the traffic with each virtual link. We have

$$\delta = \left(1 - \frac{1}{\sqrt{\Lambda}}\right)\left(1 - \frac{1}{1 + \sqrt{2}vT_\alpha}\right)$$

The results are shown in Figure 4, where the x-axis indicates Λ and y-axis indicates δ. We can see that the saving increases with increasing number of virtual links. The increase is more dramatic when the number of virtual links is small. Furthermore it also increases with increasing variability of the traffic. The more variable the traffic, the more gain we can have. When $v = 0.3$, i.e., the peak rate is about 90% more than the mean rate, the saving is more than 30% with ten virtual links. This saving is significant.

VI. CONCLUSION

By separating control plane from data plane, SDN has the potential to allow network and service providers to create a variety of new services. Virtual topology is one of the most promising services SDN can provide. Through topology abstraction process, a network provider can sell virtual slices of its physical network to different service providers. Different service providers as tenants have full control of the virtual topologies within their own slices while the network provider has the control of its physical network. However lack of information about the micro-flows of each user makes each owner difficult to explore statistical multiplexing gain.

This paper proposed VTSG as a new and practical mechanism that allows a network provider to maximize statistical multiplexing gain while still provides certain guarantee to each tenant. In addition, our VTSG approach enables network provider and service provider to make their separate resource usage decisions while sharing minimum aggregate virtual link information.

Numerical results have shown that significant gain can be achieved with relatively small number of tenants. This is very encouraging because the number of tenants a network provider will host may be variable in a very wide range.

While achieving this multiplexing gain, we have made our method for minimum bandwidth calculation to be independent from other virtual links sharing the same physical link. This feature allows all existing topology mapping algorithms applicable with our minimum bandwidth calculation method.

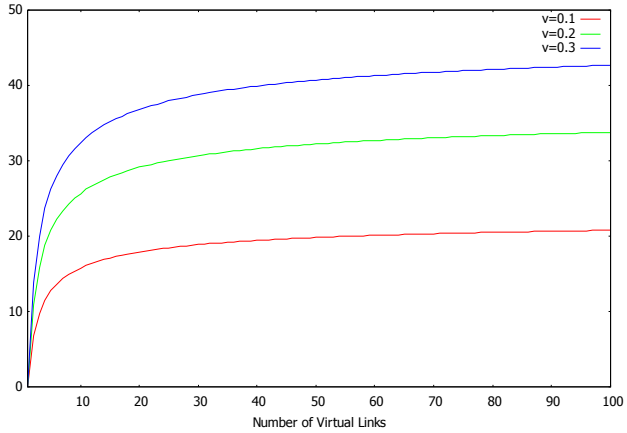

Fig. 4. Average bandwidth saved vs. minimum number of virtual links.

REFERENCES

[1] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeownm S. Shenker, "Ethane: Taking Control of the Enterprise," ACM SIGCOMM 2007, August 27-31, 2007, Kyoto.

[2] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," ACM SIGCOMM Computer Communication Review, Vil. 38, No. 2, April 2008

[3] https://www.opennetworking.org/

[4] M. Banikazemi, D. Olshefski, A. Shaikh, J. Tracey, and G Wang, "Meridian: An SDN Platform for Cloud Network Services," IEEE Communications Magazine, Feb. 2013

[5] S. Jain, et al., "B4: Experience with a Globally-Deployed Software Defined WAN," ACM SIGCOMM 2013, August 12-16, 2013, Hong Kong

[6] A. Tavakkoli et al., "Applying NOX to the Datacenter," ACM Workshop on Hot Topics in Networks (HotNets-VIII), October 22-23, 2009, New York

[7] A. Tootoonchina et al., "On Controller Performance in Software-Defined Networks," 2nd USENIX Hot-ICE'12, April 24, 2012, San Jose.

[8] A. Tootoochian and Y. Ganjali, "Hyperflow: A Distributed Control Plane for OpenFlow," USENIX NSDI INM/WREN, April 2010, San Jose

[9] S. Hassas Yeganeh and Y. Ganjali, "Kandoo: A Framework for Efficient and Scalable Offloading of Control Applications," ACM SIGCOMM HotSDN'12, August 13, 2012, Helsinki

[10] R. Sherwood, et al., "FlowVisor: A Network Virtualization Layer," OPENFLOW-TR-2009-1, OpenFlow Consortium, October 2009

[11] D. Drutskoy, E. Keller, and J. Rexford, "Scalable Network Virtualization in Software Defined Networks," IEEE Internet Computing, Vol. 17, Iss. 2, March, 2013.

[12] N.M.M.K. Chowdhury, M.R. Rahman, and R. Boutaba, "Virtual Network Embedding with Coordinated Node and Link Mapping," IEEE INFOCOM 2009, April 2009, Rio de Janeiro.

[13] X. Cheng, et al., "Virtual Network Embedding Through Topology-Aware Node Ranking," ACM SIGCOMM Computer Communication Review, Vol. 41, No. 2, April 2011.

[14] J. He, et al., "DaVinci: Dynamically Adaptive Virtual Networks for a Customized Internet," ACM CoNEXT 2008, Madrid, December 2008.

[15] J.-Y. Le Boudec and P. Thiran, "Network Calculus: A Theory of Deterministic Queuing Systems for the Internet," Springer, LNCS, 2001.

[16] W. E. Leland, M. S. Taqqu, W. Willinger, D. V. Wilson, "On the Self-similar Nature of Ethernet Traffic ," IEEE/ACM Trans. Networking, Vol. 2, No.1, Feb. 1994.