Enhancing the Performance of Post-Failure Restoration Schemes in Multi-Tenant Networks

Abdulaziz M. Ghaleb, Tarek Khalifa and Khaled Bashir Shaban
Dept. of Computer Science and Engineering, Qatar University, Doha, Qatar

***Abstract*—Failure in the physical network can cause a temporal or permanent unavailability of some resources, which can lead to a quality of service (QoS) degradation and loss of revenue. While much work has been dedicated to the survival of delay-constrained applications, little attention has been paid to enhancing the performance of the post-failure recovery scheme by maintaining the performance for the affected applications. In this paper, we introduce a QoS provisioning framework that overcomes the limitations of post-failure recovery techniques. The framework aims at not only fixing the failed applications, but also providing sufficient QoS guarantees for the hosted applications in case of link/node failure while maximizing the resources utilization. Simulation results of a data center hosting multicast applications prove that the proposed method boosts the recovery scheme to achieves better restoration ratio in a considerably fast execution time, and increases the revenue.**

***Index Terms*—Network Virtualization; Survivability; Post-Failure Recovery; QoS; Data Centers, Multi-Tenant Networks.**

I. Introduction

The failure of any component (node/link) in a substrate network hosting multiple virtual networks (VNs) can bring down multiple hosted VNs, (all those VNs that utilize the failed link/node) [1]–[4]. Typically, such failures are dealt with either pro-actively by backing up resources prior to failure or re-actively by recovering the failed VNs after failure. However, maintaining all the QoS properties and maximizing the percentage of restored VNs require extra effort in the restoration process [5]. In multi-tenant physical networks, performance guarantees of a VN starts at the embedding phase. Virtual network embedding (VNE) ensures that resources are reserved such that QoS requirements are satisfied. Notably, QoS requirements differ based on the application type and customer needs. For example, multicast applications [6]–[8] necessitate strict delay and delay variation conditions. As such, post-failure recovery algorithms should also restore the failed VNs with the QoS properties maintained. Since some resources are lost, fully recovering the failed VNs may become infeasible. However, choosing which of the VNs to restore first would affect the overall percentage of restored VNs and thus the revenue, both for which the objective is to maximize.

Enhancement of post-failure recovery schemes for better QoS provisioning during the failure recovery has never been addressed. Little contributions has been introduced to the QoS provisioning in virtual networking and cloud services with certain limitations [9], (e.g., [5], [10]–[12]). Issues pertaining to the enhancement of QoS are investigated in the form of increasing the resilience against physical failures [13], [14], and providing a QoS-aware VNE [15]. Additionally, existing recovery schemes do not adequately handle the QoS with resource awareness [3], [16]–[24]. Besides, they are mostly proactive techniques. Hence, our work differs substantially.

The contribution of this work lies on 1) introducing a framework for enhancing the post-failure recovery performance that makes the most utilization of resources, 2) conducting extensive simulations for the proposed framework for a link failure case in multicast services. The framework can be applied to any type of failure (link failure, node failure, single or multiple failures), any type of application (not exclusive for multicast applications) and any networking environment (e.g., traditional data centres, not exclusive for cloud environments). The simulation results validate the feasibility of the proposed framework and opens doors for future extensions. The framework ensures high recovery rate, high resources utilization, fast recovery, reduced penalty cost, and high revenue. Section II of the paper describes the network model and the motivation behind this work. The proposed scheme is presented in section III. The simulations and results are presented in section IV. Finally, concluding remarks are given in Section V.

II. Network Model and Problem Description

1) Substrate Network: The substrate network is represented as an undirected graph with a set of substrate nodes, N, and a set of substrate links, L, and is denoted by $G^s = (N, L)$. The bandwidth capacity of any link l ($l \in L$) is finite and is denoted by b_l, and the residual bandwidth capacity is denoted by $b_l^{'}$. Similarly, the computing capacity of any node n ($n \in N$) is finite and is denoted by c_n, and the residual capacity is denoted by $c_n^{'}$. Nodes are uniquely indexed n_i such that $n_i \in N$ and $i \in \{1, |N|\}$ where $|N|$ is the cardinality of N. Link indexing follows the nodes' indexing at the two ends of a link; $l_{i,j}$ represents a link connecting nodes n_i and n_j where $l_{i,j} \in$ L.

2) VN Request: The request for hosting a service (VNE) is defined by $G^v = (V, c', b', \Omega)$ where $V =$ is the set of virtual nodes and Ω is the constraints or QoS requirements by the service. For multicast virtual network (MVN), $V = \{s \cup T\}$ where s is a source node and T is a set of terminal nodes, and $\Omega = \{\gamma, \delta\}$ where γ and δ are delay (delay from source to terminal) and delay-variation (differential delay) constraints, respectively. Set of virtual links L' connect the source to the terminals. The bandwidth requirement $b\prime$ of each virtual link $l' \in L'$ and the computation capacity d_v of each node $v \in V$ are determined by the MVN requirements.

3) Failure impact on the performance: In Multi-tenant networks, single failure or multiple failures (failure of a set of nodes/links) can occur. Failure of any component causes unavailability of certain resources. We let $\mathbb{V}$ denote the set of

978-3-901882-85-2

embedded VNs and $\widehat{\mathbb{V}} = \{\widehat{\nu_1}, \widehat{\nu_2}, ..., \widehat{\nu_q}\}$ denote the set of affected VNs by simultaneous failures where $q \in \{1, |\widehat{\mathbb{V}}|\}$.

4) Motivation for this work: To survive the failure of physical components efficiently, a recovery process must be instantiated as transparent as possible to avoid QoS degradation and cost penalty to the infrastructure service provider (SP) while maximizing resource utilization. The amount of traffic inside data centers is four times the outbound traffic, which makes it critical as to handle the hosted applications and ensure their QoS requirements are met. Most of time-sensitive applications require resources with dedicated QoS support, however, it is difficult to grant these resources in case of failure in a multi-tenant network. Additionaly, their performance during the recovery from a failure remains crucial to customers. Failure decreases the resources availability in the network and a number of VNs can fail simultaneously due to a physical component failure. Therefore, failure causes a QoS degradation of the service due to the service disruption which results in penalty cost and loss of revenue. Moreover, as the load is high (more VNs) on an existing physical infrastructure, failures will have a greater service degradation for customers and greater penalty for the SP. Available resources have to be utilized efficiently in order to facilitate the recovery of as many failed VNs as possible. Hence, it is crucial to have a framework that prioritizes the recovery of the failed VNs.

III. Proposed scheme

An enhancement scheme for post-failure recovery is proposed to work on top of any recovery approach. It aims at improving the network resource utilization by ensuring that residual resources are not fragmented and left unused. Once failure occurs in physical resources, multiple VNs might be affected. Assuming a recovery process in one-by-one fashion, the recovery of one VN might cause a recovery blockage of another. The blocked VN could be of a higher importance or of a larger amount of revenue. The proposed framework increases the reliability of virtual networks by allowing the control/recovery of failure while optimizing the infrastructure resources by determining the order of failed VNs in the recovery process. The framework relies on several attributes, namely, the feasible region of recovery (FRR), the priority class, capacity requirements, VN size and failure impact on VNs, which are denoted as $\mathfrak{R}$, P, Ψ, Ω, and A ranking function, $F(\widehat{\nu_q})$, is designed containing the various attributes. Φ respectively. The weight of each attribute should be set according to the SP objectives, and should be adaptive to the failure impact on the network and network load. The function ranks each VN in the set of failed VNs, $\widehat{\mathbb{V}}$, and assigns an importance value to each failed VN based on specific objectives (e.g., increase restoration ratio, increase revenue, give priority to the delay-sensitive applications). They should be recovered in the order according to their importance values.

5) FRR: It indicates the region where a failed component of the failed VN can be replaced. This is determined by the afforded additional delay according to the relative position of the failure and delay requirements of the VN. The FRR of a failed VN is the region at which a backup path can be routed, or at which a terminal can be replaced. The set of or feasible regions for the affected VNs by failure is defined as $\mathbb{R} = \{\mathcal{R}_1, \mathcal{R}_2, ..., \mathcal{R}_q\}$. Recovery of one VN might cause blocking of other VN in sharing the same FRR under resource scarcity conditions. Hence, it could be more efficient to start recovery of the VN that has smaller FRR if this region is shared with other failed VNs. Fig. 1a show an example of prioritizing the recovery based on the FRR. We assume that two VNs utilize the nodes n_{14} and n_{16} through link $l_{14,16}$ to route traffic form source to terminals with each VN requiring 10 units of bandwidth, also $\widehat{\nu_1}$ and $\widehat{\nu_2}$ can afford up to 1 and 2 additional time units of delay, respectively. Hence, $\mathcal{R}_2$ is larger than $\mathcal{R}_1$, i.e., $\mathcal{R}_1 \in \mathcal{R}_2$. With a tolerance of only one additional time unit of delay, $\widehat{\nu_1}$ traffic can be re-routed from n_{14} to n_{16} through n_{11}, n_{19} or n_{20} while more choices can be obtained with two additional time units of delay. Let's consider a heavily loaded network where the only feasible route in the region is through n_{11} with a residual capacity of 15 bandwidth units. If $\widehat{\nu_2}$'s traffic is routed through n_{11}, then $\widehat{\nu_2}$'s traffic will be blocked; consequently, this VN will be terminated because of the infeasible solution. Triggering re-embedding will degrade the QoS (cause service disruption). To avoid this, VN_1 traffic should be re-routed through n_{11} since it has smaller FRR and $\widehat{\nu_2}$ traffic can be re-routed through other nodes such as n_6 and n_7 (Fig. 1a). Similarly, given that n_8 is hosting terminal nodes (Fig. 1b) for $\widehat{\nu_1}$ and $\widehat{\nu_2}$ denoted as $t_{\widehat{\nu_1}}$ and $t_{\widehat{\nu_2}}$, respectively. If n_8 is brought down by a failure of itself or a failure of $l_{12,8}$, $t_{\widehat{\nu_1}}$ and $t_{\widehat{\nu_2}}$ should be migrated to other host(s). Let's consider the case where storage is available while computation requirements for either $\widehat{\nu_1}$ or $\widehat{\nu_2}$ can be satisfied by the n_{13} and n_{23}. Migrating $t_{\widehat{\nu_2}}$ to n_{13} will block the migration of $t_{\widehat{\nu_1}}$ and cause the termination of VN_1. Hence, it is better to consider the recovery of the failed VN with smaller FRR that is $F(\widehat{\nu_q}) \propto \mathfrak{R}_q$ where $\mathfrak{R}_q$ is the FRR of $\widehat{\nu_q}$.

6) Priority class: Priority defines actions applied to a modular QoS policy. For example, a priority feature allows a specified amount of delay-sensitive traffic (higher priority) to be sent before other traffic. Higher priority requires higher availability of resources, and hosted applications with higher priority agreement brings more revenue to the SP. Besides, failure to deliver the service will cause high penalty cost and affects the network reliability. Hence, priority class should be integrated into the recovery model (higher-priority VNs should be recovered first) in order to provide better post-failure QoS and maximize revenue, thus, $F(\widehat{\nu_q}) \propto P_q$ where P_q is the priority class of $\widehat{\nu_q}$.

7) Capacity requirements: They refer to any combination of the links bandwidth, nodes computation, and storage requirement. An efficient capacity management approach should be considered during the embedding process in order to avoid resource under-utilization and to satisfy customers due to the resources unavailability. Furthermore, capacity management should be adopted for failure recovery process to maintain good QoS level by providing fast failure recovery, minimizing VNs termination and optimize resource utilization by utilizing

residual resource that can be left unused due to bad capacity management, hence, maximizing the SP revenue.

In this scheme, a failed VN with the requirements of a higher link bandwidth, higher computation capacity, and more storage should be recovered first though it can not be the optimal approach always. Let's consider a link failure scenario as in Fig. 1c. Node n_1 hosts the virtual source nodes $s_{\widehat{\nu_1}}$, $s_{\widehat{\nu_2}}$ and $s_{\widehat{\nu_3}}$; and n_6 hosts the virtual terminal nodes $t_{\widehat{\nu_1}}$, $t_{\widehat{\nu_2}}$ and $t_{\widehat{\nu_3}}$ for $\widehat{\nu_1}$, $\widehat{\nu_2}$ and $\widehat{\nu_3}$, respectively. $\widehat{\nu_1}$ requires 8 bandwidth units while $\widehat{\nu_2}$ and $\widehat{\nu_3}$ require 4 bandwidth units. Every link has a residual capacity of 10 bandwidth units. When failure brings down $l_{2,6}$, traffic for the three VNs should be re-routed through n_3 or n_5. However, if $\widehat{\nu_3}$ traffic is re-routed through n_3 and $\widehat{\nu_2}$ traffic through n_5, the residual bandwidth capacities of all links are 6 bandwidth units. Hence, $\widehat{\nu_3}$ can not be re-routed through any path. If $\widehat{\nu_1}$ is re-routed through either n_3 or n_5, $\widehat{\nu_2}$ and $\widehat{\nu_3}$ can be re-routed through the other node.

In Fig. 1d, node migration in needed due to the failure of n_1 that hosts terminal nodes for $\widehat{\nu_1}$, $\widehat{\nu_2}$ and $\widehat{\nu_3}$ denoted as $t_{\widehat{\nu_1}}$, $t_{\widehat{\nu_2}}$ and $t_{\widehat{\nu_3}}$, respectively. Assuming that n_5 and n_7 are the only available hosts for migrations, $t_{\widehat{\nu_1}}$, $t_{\widehat{\nu_2}}$ and $t_{\widehat{\nu_3}}$ should be migrated to them. Given that $t_{\widehat{\nu_1}}$ requires 8 virtual CPUs (vCPUs) for computation while each of $t_{\widehat{\nu_2}}$ and $t_{\widehat{\nu_3}}$ require 4 vCPUs, and each of n_5 and n_7 has available 10 vCPUs. Similarly, if $t_{\widehat{\nu_2}}$ is migrated to one node and $t_{\widehat{\nu_3}}$ is migrated to the other, it will be impossible for $t_{\widehat{\nu_1}}$ since the residual computation capacity in both nodes is 6 vCPUs while $t_{\widehat{\nu_1}}$ requires for 8 vCPUs. Contrarily, if $t_{\widehat{\nu_1}}$ is migrated first, $t_{\widehat{\nu_2}}$ and $t_{\widehat{\nu_3}}$ can be migrated too. Hence, $F(\widehat{\nu_q}) \propto \Psi_q$ where Ψ_q is the capacity requirement of $\widehat{\nu_q}$.

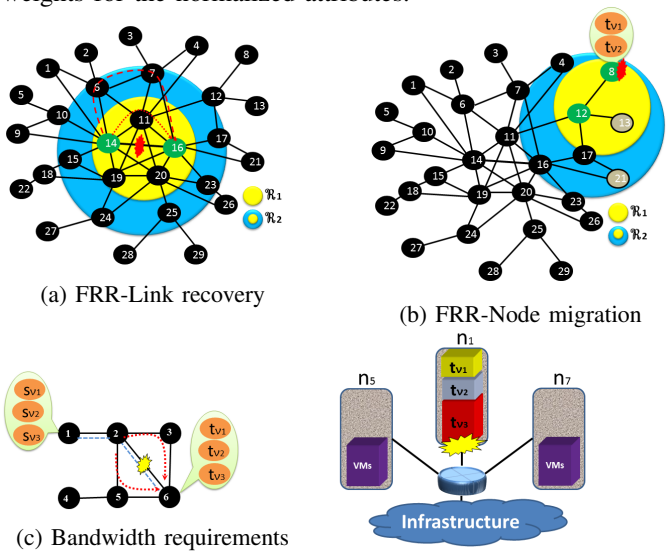
8) VN Size: It is more profitable to consider the recovery of larger VNs first as discussed in section III-7, in case if failure of some nodes would bring down the whole VN. Assuming a failure of two VNs, $\widehat{\nu_1}$ and $\widehat{\nu_2}$ with 20 and 5 virtual nodes, respectively, and with the same capacity requirement and same FRR. If one virtual node is affected in both VNs and resource is available to the recovery of one NV, the recovery of $\widehat{\nu_1}$ and release of $\widehat{\nu_2}$ is the wise decision from revenue prospective. Hence, $F(\widehat{\nu_q}) \propto \Omega_q$ where Ω_q is the size of $\widehat{\nu_q}$.

9) Failure impact: The number of affected components of failed VN impacts the restoration probability and its time. Therefore, VNs with lighter failure impact, Φ, should be restored first since they are more likely to be recovered and recovered quickly. This will allow for the restoration of more failed VNs. This implies that $F(\widehat{\nu_q}) \propto 1/\Phi_q$ where Φ_q is the failure impact on $\widehat{\nu_q}$. Failure impact on the physical network is reflected as resource unavailability, which affects the restoration probability especially in a highly loaded network.

The objectives of can vary based on the SP need. For example, they could be to recover more failed VNs, increase revenue and/or give priority to the delay-sensitive applications. Alternatively, the provider could focus on recovering higher-priority VNs first or could prioritize higher revenues VNs. The ranking function is related to the attributes as follows.

$$F(\widehat{\nu_q}) = \alpha\overline{\Re_q} + \beta\overline{P_q} + \epsilon\overline{\Psi_q} + \varepsilon\overline{\Omega_q} + 1/\eta\overline{\Phi_q} \qquad (1)$$

where $\overline{\Re_q}, \overline{P_q}, \overline{\Psi_q}, \overline{\Omega_q}$ and $\overline{\Phi_q}$ are the normalized values of $\Re_q, P_q, \Psi_q, \Omega_q$ and Φ_q, respectively. They are calculated as the attribute values of $\widehat{\nu_q}$ over the maximum value of the attribute in $\widehat{\mathbb{V}}$ denoted as $\Re_{\max}, P_{\max}, \Psi_{\max}, \Omega_{\max}$ and $\Phi_{\max}$. The values of $\alpha, \beta, \epsilon, \varepsilon$ and η are $\in \{0, 1\}$ and represent the weights for the normalized attributes.



(a) FRR-Link recovery

(b) FRR-Node migration

(c) Bandwidth requirements

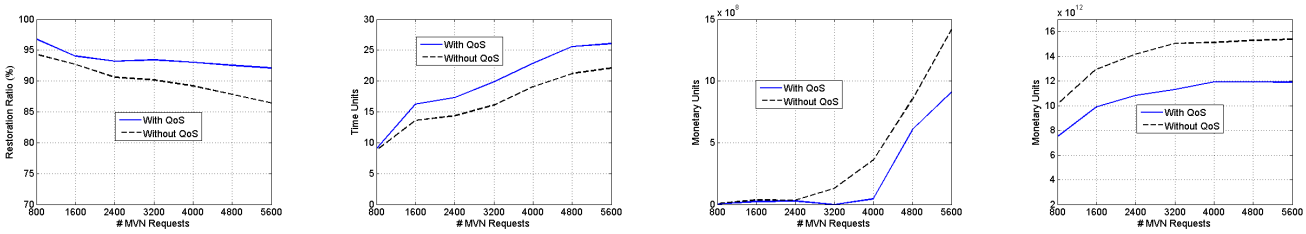
(d) Computation requirements

Fig. 1: Illustrative Examples

IV. Experimental Work

The experiments assume a network that hosts multicast applications (MVNs), which require strict delay and delay variation between the terminals (8-20 terminals). We have adopted the 3-step multicast VNE process in [25] and the link failure recovery algorithms in [26] and a K-ary FatTree network topology (K=16). The substrate network contains $K^3/4$ servers (each server with 64 vCPUs), 64 core switches, 128 aggregate switches, 128 access switches and total of 3072 links (each with 10 Gbps) connecting the different nodes (core switches, aggregation switches, access switches, and servers). We assume that storage is always enough but the focus is on the links bandwidth in Gbps (0.1-1) and CPU computation requirements in terms of the number of vCPUs (1-8 for each virtual node). The enhancement scheme is applied to rank the failed MVNs; an MVN with higher rank is to be recovered first. The ranking depends on the affordable delay (FRR), the priority class, link bandwidth (capacity requirements) since the recovery methods include finding alternative paths and the MVN size. The values of $\alpha, \beta, \epsilon, \varepsilon$ and η in Eq. (1) have been set such that they have equal importance. We evaluate the performance at different network loads (# MVN requests) using restoration ratio, average execution time, penalty cost and total revenue. Results were collected twice (same simulation setup): one experiment is with our QoS-aware recovery method and the second experiment is without.

Fig. 2a depicts restoration ratio, total number of recovered MVNs to the total number of MVNs brought down by the failure. It measures the reliability of the recovery schemes. The results show that the proposed scheme increases the



(a) Restoration ratio (b) Average execution time (c) Penalty Cost (d) Total revenue

Fig. 2: Simulation results

reliability of the recovery method. This will directly affect the QoS provisioning since more MVNs are recovered. The trend shows that the scheme has a higher impact on the restoration ratio during the higher-load conditions, which is expected. Fig. 2b shows the average execution time (over the number of recovered MVNs) versus the network load. The time here is the search time for alternative paths and the time for constructing the multicast tree in addition to the time used by the proposed framework to rank the failed MVNs. There is a slight increase in the execution time (in the simulator) due to applying the proposed scheme, which in practice is negligible. Furthermore, the proposed method recovers more large-in-size MVNs which increase the recovery time in the adopted link recovery scheme [26]. The difference is expected to be negligible for other recovery scenarios comparing to practical scenarios. The penalty cost is compared for the link recovery scheme with and without QoS implementations in Fig. 2c. The graphs show a lower penalty cost when deploying the QoS scheme, which is translated as an enhancement in revenue. The lower penalty cost associated with QoS implementations is due to higher restoration ratio, restoration of higher priority MVNs first, and the restoration of larger MVNs first. Fig. 2d shows the total revenue. More revenue is obtained due to recovering the more expensive MVNs (higher priority), recovering the larger MVNs, and increasing the admittance ratio.

V. Conclusion

We conclude that QoS enhancement schemes are important for efficient network management. They have good impact on the performance under low and high load conditions. Future work includes evaluation of the proposed scheme in different recovery algorithms, in different multi-failure environments and with different network topologies.

Acknowledgment

This was made possible by NPRP 5-137-2-045 grant from the Qatar National Research Fund (The Qatar Foundation). The statements made herein are solely the responsibility of the authors.

References

[1] A. Fischer, J. Botero, M. Till Beck, H. de Meer, and X. Hesselbach, "Virtual network embedding: A survey," *IEEE Comm. Surveys Tutorials*, vol. 15, no. 4, pp. 1888–1906, 4th 2013.

[2] T. A., L. P., S. S., and J. T., "Overcoming the internet impasse through virtualization," *Computer*, vol. 38, no. 4, pp. 34–41, April 2005.

[3] M. Rahman and R. Boutaba, "Svne: Survivable virtual network embedding algorithms for network virtualization," *IEEE TNSM*, vol. 10, no. 2, pp. 105–118, June 2013.

[4] S. Herker, A. Khan, and X. An, "Survey on survivable virtual network embedding problem and solutions," in *9th Int. Conf. on Networking and Services*. IARIA, 2013, pp. 99–104.

[5] I. Ayadi, N. Simoni, and G. Diaz, "Naas: Qos-aware cloud networking services," in *12th IEEE NCA*, Aug 2013, pp. 97–100.

[6] Y. Miao, Q. Yang, C. Wu, M. Jiang, and J. Chen, "Multicast virtual network mapping for supporting multiple description coding-based video applications," *Comp. Net.*, vol. 57, no. 4, pp. 990 – 1002, 2013.

[7] M. Zhang, C. Wu, M. Jiang, and Q. Yang, "Mapping multicast service-oriented virtual networks with delay and delay variation constraints," in *IEEE GLOBECOM*, Dec 2010, pp. 1–5.

[8] A. Iyer, P. K., and V. M.,"Avalanche:Data center multicast using software defined networking," *6th Int. Conf. on COMSNETS*, Jan 2014, pp. 1–8.

[9] P. Rygielski and S. Kounev, "Network virtualization for qos-aware resource management in cloud data centers: A survey," *PIK*, vol. 36, no. 1, pp. 55 – 64, 2013.

[10] G. Stavrinides and H. Karatza, "A cost-effective and qos-aware approach to scheduling real-time workflow applications in paas and saas clouds," in *3rd Int. Conf. on FiCloud*, Aug 2015, pp. 231–239.

[11] M. Salama, A. Zeid, A. Shawish, and X. Jiang, "A novel qos-based framework for cloud computing service provider selection," *Int. J. Cloud Appl. Comput.*, vol. 4, no. 2, pp. 48–72, Apr. 2014.

[12] I. Ayadi, G. Diaz, and N. Simoni, "Qos-based network virtualization to future networks: An approach based on network constraints," in *4th Int. Conf. on NOF*, Oct 2013, pp. 1–5.

[13] X. Zhang, C. Phillips, and X. Chen, "An overlay mapping model for achieving enhanced qos and resilience performance," in *3rd Int. Cong. on ICUMT*, Oct 2011, pp. 1–7.

[14] J. Shamsi and M. B., "Qosmap: Achieving quality and resilience through overlay construction," *4th Int. Conf. on ICIW*, May 2009, pp. 58–67.

[15] A. J. and A. K., "Periodical auctioning for qos aware virtual network embedding," *IEEE 20th Int. Workshop on IWQoS*, June 2012, pp. 1–4.

[16] H. Yu, V. Anand, C. Qiao, and G. Sun, "Cost efficient design of survivable virtual infrastructure to recover from facility node failures," in *IEEE ICC*, June 2011, pp. 1–6.

[17] C. Q., B. Guo, S. H., J. W., T. W., and W. Gu,"A novel 2-step approach to surviving facility failures," *OFC/NFOEC*, Mar 2011, pp. 1–3.

[18] H. Yu, C. Qiao, V. Anand, X. Liu, H. Di, and G. Sun, "Survivable virtual infrastructure mapping in a federated computing and networking system under single regional failures," *IEEE GLOBECOM*, Dec 2010, pp. 1–6.

[19] J. Xu, J. Tang, K. Kwiat, W. Zhang, and G. Xue, "Survivable virtual infrastructure mapping in virtualized data centers," *Proc. of IEEE Int. Conf. on Cloud Comput.*, IEEE Comp. Society, 2012, pp. 196–203.

[20] Q. Hu, Y. Wang, and X. Cao, "Location-constrained survivable network virtualization," in *35th IEEE SARNOFF*, May 2012, pp. 1–5.

[21] A. M. Ghaleb, T. Khalifa, S. Ayoubi, K. Shaban, C. Assi, "Surviving Multiple Failures in Multicast Virtual Networks with Virtual Machines Migration," in *IEEE TNSM*, vol.PP, no.99, 2016.

[22] S. Ayoubi, Y. Chen, and C. Assi, "Pro-red: Prognostic redesign of survivable virtual networks for cloud data centers," in *26th Int. ITC*, Sept 2014, pp. 1–9.

[23] T. Guo, N. Wang, K. M., and R. T., "Shared backup network provision for virtual network embedding," *IEEE ICC*, June 2011, pp. 1–5.

[24] H. Yu, V. Anand, C. Qiao, and H. Di, "Migration based protection for virtual infrastructure survivability for link failure," in *OFC/NFOEC*, March 2011, pp. 1–3.

[25] S. Ayoubi, C. Assi, K. Shaban, and L. Narayanan, "Minted: Multicast virtual network embedding in cloud data centers with delay constraints," *IEEE Tran. on Comm.*, vol. 63, no. 4, pp. 1291–1305, April 2015.

[26] A. M. Ghaleb, T. Khalifa, S. Ayoubi, and K. Shaban, "Surviving link failures in multicast vn embedded applications," in *NOMS*,Apr 2016.