New Fault-Tolerant Datacenter Network Topologies

Rana E. Ahmed and Heba Helal
Department of Computer Science and Engineering, American University of Sharjah, Sharjah, United Arab Emirates
Email: rahmed@aus.edu; g00065362@aus.edu

Abstract — Data Center Network (DCN) topologies, being integral parts of cloud computing infrastructure, are vulnerable to failures due to the presence of huge number of servers, switches and links. One of the biggest challenges in the fault-tolerant DCN design is to provide a graceful degradation in performance in the event of a link or server failure. This paper proposes two new fault-tolerant DCN topologies, called DCell-Star and DCell-Ring. The proposed topologies are cost-effective, scalable, fault-tolerant, and enhance overall performance (throughput and latency) of the standard DCell topology. Performance evaluation of the proposed topologies is done through a simulation study, and the results are compared with the standard DCell topology. The comparison is done in terms of various metrics such as throughput, latency, and diameter. The simulation results show that the proposed topologies outperform the standard DCell topology due to the availability of multiple alternate shortest paths between pairs of servers, resulting in an improvement of about 5% in throughput even for a small-size network.

Index Terms—Data center networks, network topologies, fault-tolerance, performance evaluation

I. Introduction

Due to tremendous growth in cloud-based services, such as Google Search and Facebook social network, the overall cloud infrastructure is constantly evolving to support a huge volume of data and QoS requirements of such services. Data Center Network (DCN) infrastructure consists of a massive number of servers with large online services. A DCN topology connects those servers through switches and links with high capabilities which, in turn, connects to the external world. The performance of DCN can be estimated using metrics such as, throughput, latency, bandwidth, power consumption, reliability, cost, etc. A number of data center network architectures have been proposed, which are classified into two major categories: fixed-topology, and flexible-topology. For the fixed-topology architecture, the network topology cannot be changed after the network is deployed, while the change is possible in the flexible-topology architectures. Some example architectures for fixed-topologies are: Fat-tree [1], Portland [2], and recursive topologies such as DCell [3] and BCube [4]. The flexible topologies, using optical switching technology, include c-Through [5]. Another way of classification of DCN topologies is based on whether they are server-centric or switch-centric networks. In server-centric network, the addressing and routing tasks are performed by the servers. Both DCell and BCube topologies are the examples of server-centric networks. On the other hand, all routing and addressing tasks are done by switches and routers in case of switch-centric networks. Fat-tree and Portland architectures belong to this category.

With ever increasing data traffic in clouds, there has been a great interest in improving the performance of cloud infrastructure. Data centers offer huge computational power and storage, reliability, availability, and cost-effective solutions needed by the cloud applications. As a data center network topology consists of a very large number of servers (hosts), switches, routers and communication links, the topology is vulnerable to failures (permanent or transient). In order to provide high availability and a certain acceptable performance level, the network topology must be designed to recover from such failures. At the physical hardware level, redundant links and switches present in the topology may aid in providing such recovery (or fault-tolerance). Moreover, path selection algorithm for the topology may offer alternate paths (if they exist) between hosts (servers) that will be available in the event of a switch or link or server failure.

There has been some recent research interest in improving the performance of DCN topologies to provide higher throughput, lower latency and better fault-tolerance. The improvements can be achieved either by modifying the architecture of the topology such that it increases the utilization of available bandwidth and decreases the latency, or by adding extra links between devices to enhance the fault-tolerance. Another practical way of providing fault-tolerance is by introducing new path selection algorithms that provide alternate paths in the network in case of failures in links, server and switches.

Several datacenter network topologies have been proposed and implemented in real data centers. The list includes Google Fat-Tree, DCell, BCube, and Facebook Fat-Tree. A good survey about DCN topologies can be found in [6], [7]. Most of the earlier research work carried out deals with the performance evaluation of topologies without consideration of any failure in the network topology. Moreover, different types of simulators with different traffic generators were used to evaluate the performance. Therefore, the comparisons

Manuscript received January 12, 2018; revised May 15, 2018.
This work was supported by the American University of Sharjah, UAE, for providing a research grant (FRG16-R-13)
Corresponding author email: rahmed@aus.edu
doi:10.12720/jcm.13.6.259-265

among topologies become difficult on the similar baseline parameters. In [8], two DCN topologies, Fat-Tree and DCell, are compared using random traffic with exponential and uniform distribution. Simulation results from ns-3 simulator show that the performance of Fat-Tree is better than DCell in terms of throughput and latency. In [9], an evaluation of the fault-tolerance characteristics of some DCN topologies is made using a Java-based simulator. Several metrics such as Average Path Length (APL), Aggregate Bottleneck Throughput (ABT) and connection failure ratio are analyzed. Fault-tolerance capabilities are estimated/ analyzed using several metrics for four different DCN topologies (Fat-Tree, DCell, BCube, and FlatNet) in [10]. Simulator DCNSim is used to capture the effects of various failures in the topologies in details in terms of connection-oriented metrics which include ABT, APL and Routing Failure Rate (RFR). Several popular architectures are compared with respect to various parameters, such as power, cost, scalability, hop counts and path diversity in [11] using analytical methods. The "Mininet" simulator is used in [12] for the evaluation of throughput; however, results regarding the fault-tolerance characteristics of the topologies are not considered. The work in [13] reports a new topology, called Diamond, which improves fault tolerance capabilities of the Fat-Tree topology. The Diamond topology replaces all aggregate switches with edge switches. With this new Diamond topology, the average path length and end-to-end delay decrease by 10% compared with the original Fat-Tree topology. The Diamond topology also proposes a routing algorithm, called Fault-Avoidance Routing (FAR), with the ability of handling thousands of servers and providing fault-tolerance. Furthermore, [14] proposes a server-centric, DCN topology, named BCube Connected Crossbars (BCCC). The performance of the topology is found to outperform BCube under some metrics such as network diameter, performance in the event of server/link failure, expandability and server port utilization [15].

This paper presents two new fault-tolerant, scalable, and high-throughput DCN topologies. The fault-tolerance in the proposed topologies exploits both hardware redundancy and multipath routing.

The rest of this paper is organized as follows: Section II introduces the structure of the two proposed DCN topologies. Section III describes the simulation environment used, while Section IV presents the results of performance comparison of the proposed topologies with the well-known topologies. Finally, Section V provides the conclusions.

II. Proposed Data Center Network Topologies

In this section, we propose two new DCN topologies that are derived from the standard DCell topology with some modifications. We have added a few extra switches and links aiming to improve the performance in terms of throughput, latency and fault-tolerance. We also discuss how to construct new topologies recursively.

A. DCell-Star

Fig. 1 shows the structure of the proposed topology called DCell-Star. In this topology, an additional (central) switch is added at the higher level to connect together all the mini-switches present at lower levels. The main reasons of providing the central switch are to provide multiple alternate paths between hosts, and to reduce the diameter and the average shortest path length. Fig. 1 shows an example of DCell-Star$_1$ structure which consists of five basic DCell-Star$_0$ units. Each DCell-Star$_0$ consists of n = 4 servers and a mini-switch connecting them together. For constructing level-1 of DCell-Star (i.e., DCell-Star$_1$), (n + 1) DCell-Star$_0$ units are connected through dual-port servers, and the central switch connects all mini-switches. Each mini-switch needs one more port for its connection with the central switch, resulting in a total of (n + 1) ports. The central switch requires (n+2) ports; as (n + 1) ports are needed to connect all mini-switches and one port for the connection to the higher level central switch.

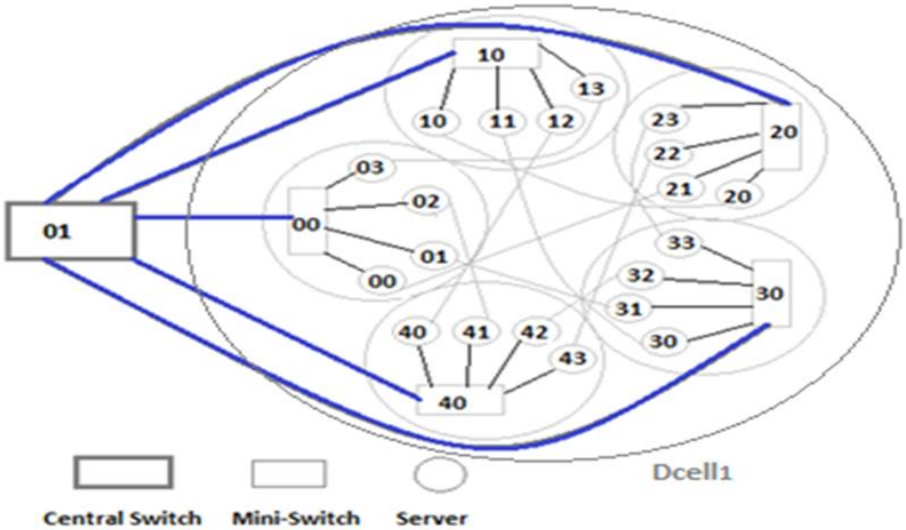

Fig. 1. DCell-Star$_1$ topology

Fig. 2 shows the structure of DCell-Star$_2$ following similar steps in constructing topology using DCell-Star$_1$ units with n =2, where n represents the number of server in a basic DCell-Star$_0$ unit.

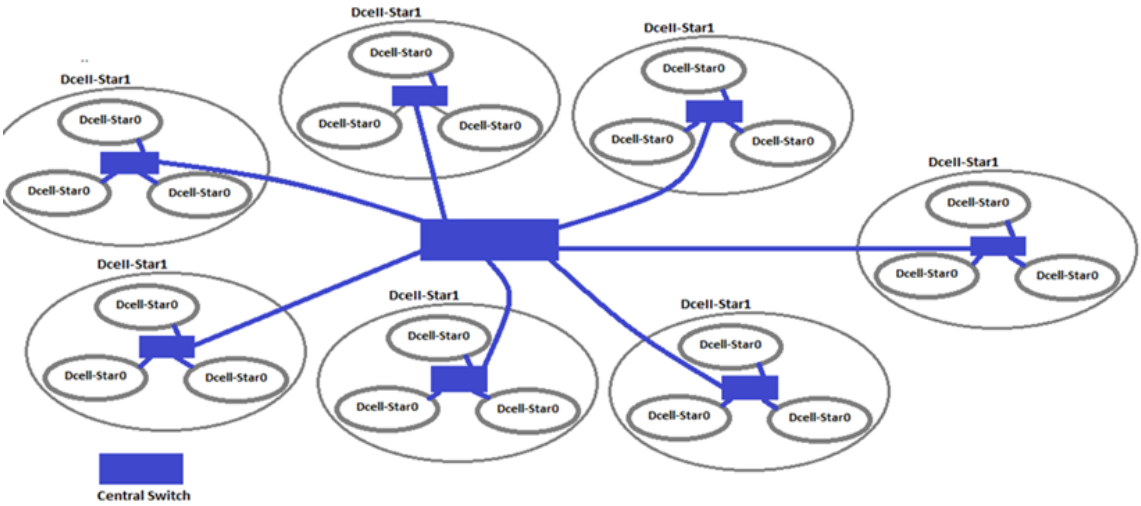

Fig. 2. DCell-Star$_2$ topology with n = 2.

The address of a server in DCell-Star$_k$ unit can be denoted as $\mathbf{a_t\ a_{t-1}\ \ldots a_1\ a_0}$, where $\mathbf{a_0}$ refers to the index of the server in DCell-Star$_0$ and it takes values from 0 to (n-1), and $\mathbf{a_i}$ represents unique ID for the DCell-Star$_0$ unit, with $1 \le i \le k$. We can say that the two hosts A and B with addresses $\mathbf{a_t\ a_{t-1}\ \ldots a_1\ a_0}$ and $\mathbf{b_t\ b_{t-1}\ \ldots b_1\ b_0}$, respectively, are in the same lowest level DCell-Star$_0$ when $\mathbf{a_i = b_i}$ and $\mathbf{a_0 \neq b_0}$, for $1 \le i \le k$. Following a similar approach, the address of a mini-switch can be represented as $\mathbf{s_t\ s_{t-1}\ \ldots s_1\ s_0}$ where $\mathbf{s_0 = k}$ indicates the next lower level of DCell-Star$_k$ and $\mathbf{s_i}$ (with $1 \le i \le k$) represents the unique ID for DCell-Star$_0$.. The addressing scheme of the central switch is exactly the same as used with mini-switch, except that $\mathbf{s_0 (= k)}$ represents the level of DCell-Star$_k$.

As mentioned earlier, two servers are in the same DCell-Star$_k$ when all digits in their addresses are equal except the least significant one. Therefore, forwarding the packets between two servers in the same DCell-Star$_0$ is done through the related mini-switch with $\mathbf{s_i = a_i}$. By considering one hop count between any two directly connected devices, the hop count between two hosts in the same DCell-Star$_0$ will be two. However, when two servers are located in different DCell-Star$_0$s, packets are first routed to the related mini-switch, then to the central switch which in turns directs the packets to the mini-switch connected to the destination node. This path needs a hop count of 4 to reach to the desired destination server.

B. DCell-Ring

The second proposed topology, DCell-Ring, also modifies the standard DCell topology, and all min-switches at the same level are connected together, forming a ring as shown in Fig. 3. The main objectives of linking the mini-switches together in the form of a ring are to provide multiple alternate paths between hosts, decrease the average shortest path length and the diameter of the standard DCell topology. Fig. 4 displays the structure of DCell-Ring$_2$ following the similar steps in constructing topology using DCell-Ring$_1$ units with n = 2.

The addressing mechanisms for servers and mini-switches are exactly the same as formulated for DCell-Star. For the routing, transmitting the packets within the same DCell-Ring$_0$ is done via the related mini-switch, with a hop count of 2. However, when the servers are in different DCell-Ring$_0$s, the packets are first routed to the related mini-switch, then routed towards the desired destination mini-switch through the ring, and finally to the destination server. In this case, we need a hop count of at least 3 when two hosts are in two different DCell-Ring$_0$s and the mini-switches are directly connected.

III. Simulation of DCN Topologies

A. Simulation Parameters

We evaluated the performance of various well-known and the proposed topologies via simulation study. The Google Fat Tree topology considered for the simulation consists of three levels: core, aggregate and edge; while at the fourth level we have hosts connected. Each edge switch connects to two hosts with a total of 16 hosts. The number of hosts to be connected is limited by the number of ports present in the switch.

The standard DCell topology is simulated for level 1 (i.e., DCell$_1$). It consists of five mini-switches with four hosts connected to each mini-switch. There is one link between two hosts present in different DCell$_0$s.We encounter a problem in forwarding the packets from a host having two different network interfaces (ports). Using an approach used in [25], we solve this problem by introducing a combo-switch connected to each host. This switch allows the two interfaces attached to the server to be used simultaneously for sending packets. In addition, the two proposed topologies (i.e., DCell-Star and DCell-Ring) are simulated with the same set of parameters.

The BCube topology is simulated for level 1 with core and edge switches. Each edge switch is attached to four hosts with a total of 16 hosts.

The Facebook Fat-tree topology is simulated using four fabric switches and 16 top of rack switches (TORs). Each TOR switch connects only to one host, and we have a total of 16 hosts.

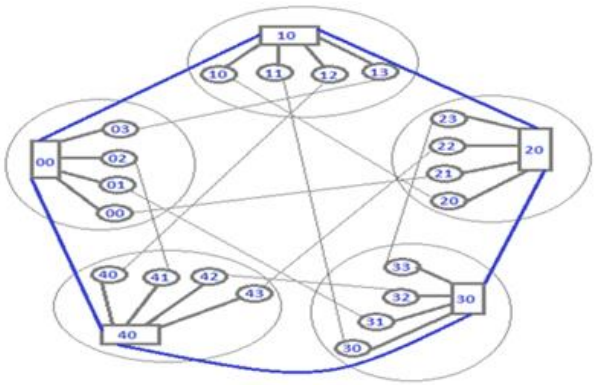

Fig. 3. DCell-Ring$_1$ topology

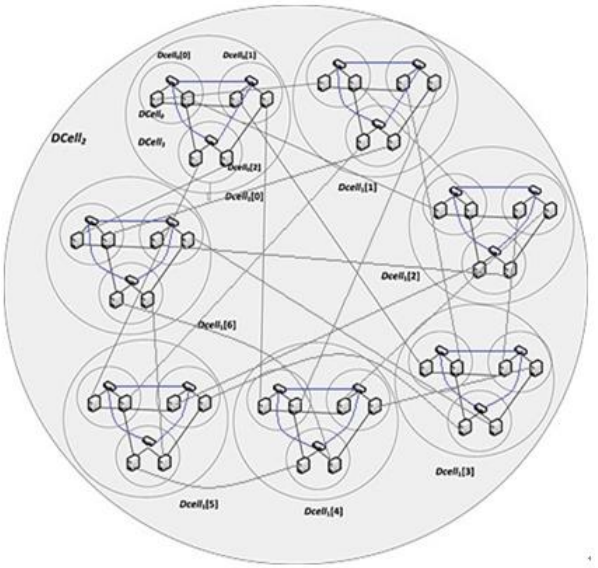

Fig. 4. DCell-Ring$_2$ topology with n =2.

B. Datacenter Traffic Patterns

We used different traffic models reported in [26] to inject traffic in DCN to evaluate the performance of topologies. These models are representatives of inter-data center traffic scenarios. The following traffic models are used:

- *Uniform Random traffic*: Each host sends traffic to any random host with the same probability.
- *Stride Traffic*: A host with an index i sends packets to host with an index (i + (N-1)/2) mod N, where N is the total number of hosts.
- *Bit Complement Traffic*: Each node sends traffic to another node with an index of bitwise inversion of the sender node.
- *One-to-All:* One host sends packets to all other hosts and the results are averaged.
- *All-to-All*: All hosts send packets to all other hosts. This traffic model represents the highest traffic loads applied to the network.

C. The "Mininet" Simulator

The Mininet simulator [12] provides network analysis with a realistic virtual network of nodes, switches and links. It can create a network topology with simple shell commands (e.g., sudo mn), and a large network topology with several parameters (e.g., link bandwidth, link delay, etc.) can be programmed using python API. All network elements and flows are controlled by POX controller [27]. The controller is initialized in VirtualBox and it helps in communicating switches to each other via OpenFlow protocol [28] which controls the forwarding of the routing tables remotely. The selection of this specific controller with respect to the other available ones (e.g. RIPLPOX, OpenDaylight, and Floodlight) is based on its demonstrated consistency in the network coverage. The traffic is transmitted from the sender to the controller which, in turn, sends it to the destination according to the information in the flow table. Moreover, STP (Spanning Tree Protocol) [22] is enabled in the network to avoid infinite loops in multi-path topologies.

In addition, RYU controller [29] can be used for setting static and default routes for each router in the network topology to forward the packets among the hosts. It has a capability to let a switch with a unique ID to join the topology as a router. This is beneficial especially when we want the message to follow a certain path under some required specifications like minimum hop count, low latency, shortest path, etc.

IPerf [30] and Ping [31] commands are used to measure throughput and latency, respectively. IPerf estimates the bandwidth consumed between any two hosts using Transmission Control Protocol (TCP) or User Datagram Protocol (UDP), while Ping has the capability to measure the latency of packet flow, percentage of packet loss, jitter, and to specify the size of transmitted packets. All tests are run on Intel Core i5-5200U CPU @ 2.2GHz with 6G RAM. We simulated the topologies with and without failures and the results are presented in the next section.

IV. Results and Discussion

In this section, we compare the standard DCell topology with the two proposed topologies with a variety of parameters such as throughput, latency, fault-tolerance, diameter, and average shortest path length.

A. Throughput

Fig. 5 shows the simulation results for the normalized throughputs of the two proposed topologies DCell-Star and DCell-Ring, and the results are compared with the standard DCell topology under different traffic models. We can conclude, from the figure, that DCell-Star and DCell-Ring topologies have higher throughputs compared to the standard DCell topology. In addition, DCell and D-

Cell-Ring topologies show relatively close results. Increase in throughput in the proposed topologies is mainly due to the larger number of the shortest paths that potentially reduces the traffic congestion by transmitting the traffic in parallel.

With an increasing value for n (the number of hosts in the basic DCell-Star$_0$ or DCell-Ring$_0$ structure), the total number of hosts in the topology increases. With an increase in n, we observe higher percentage of improvements in the throughput results for both proposed topologies compared with the standard DCell. As shown in Fig. 6, the throughput achieved in two proposed topologies exceeds that in the standard DCell topology for almost all traffic models. However, the increase in the throughput in DCell-Star is higher than the DCell-Ring for all traffic scenarios.

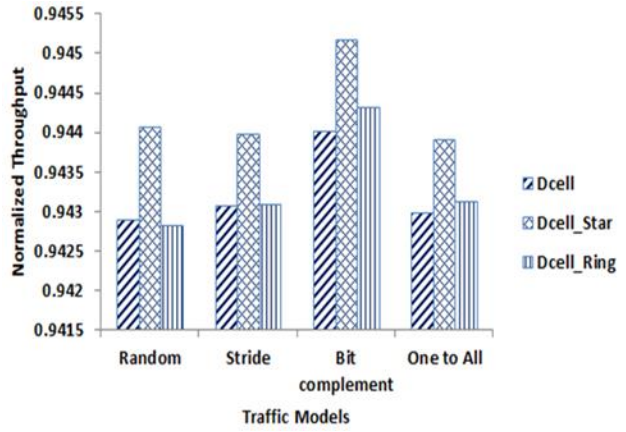

Fig. 5. Throughput comparison for n = 4.

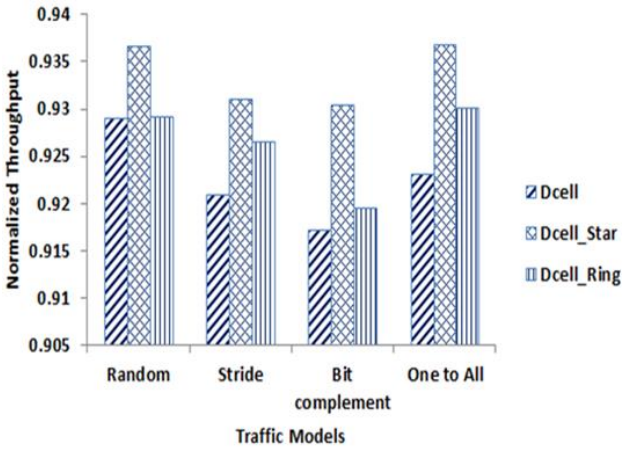

Fig. 6. Throughput comparison for n = 16.

B. Latency

Fig. 7 summarizes the simulation results for the average packet latencies. The proposed topologies have lower latencies as compared to the standard DCell due to their lower values of diameters.

C. Fault-Tolerance

In order to study the effect of failures in two new proposed network topologies, we simulate each topology and monitor the behavior of the topology under two types of failure: transient and permanent. We simulate a link and a switch failure. Due to the presence of the parallel shortest paths in DCell-Star and DCell-Ring topologies, both achieve better fault-tolerance characteristics as compared to the standard DCell topology. Transient and permanent failures generate similar behavior, as shown in Fig. 8 in the case of One-to-One. The transient failure is simulated to be present in the system for 5 msec and the behavior is monitored at the interface (switch port) connected towards the lower level in the topology. When the controller notices any failure in the link or switch, it chooses the next shortest path available between any two hosts. The standard DCell topology shows similar results, as shown in Fig. 9, but with a slightly lower throughput. The proposed topologies outperform DCell topology in the utilization of most links. In the standard DCell, there are more ripples (sudden changes) in the throughput over time as compared to the proposed topologies, as shown in Fig. 8.

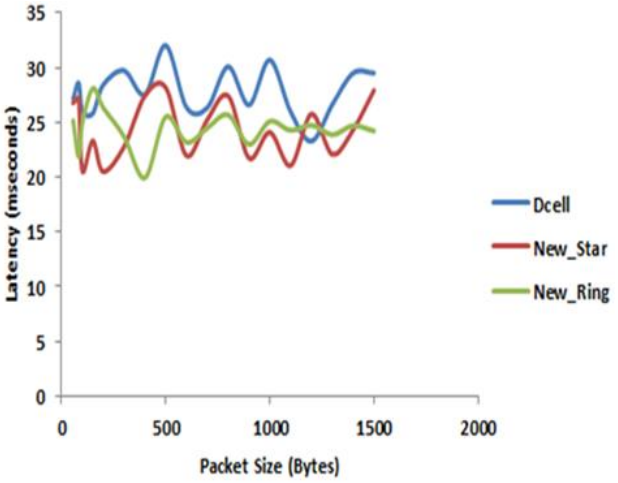

Fig. 7. Average latency vs packet size (All-to-All traffic model).

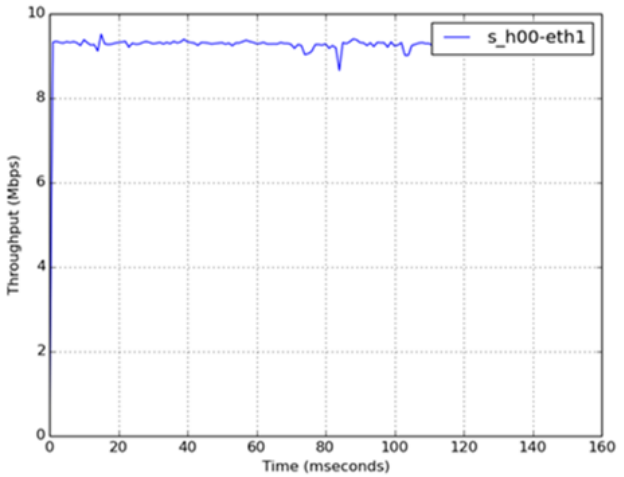

Fig. 8. Throughput under Transient failures in the proposed topologies (One-to-One traffic).

D. Discussion on Results

From the simulation study, it is clear that the standard DCell topology performs the worst in terms of throughput and latency metrics as compared to the proposed

topologies. We can improve the performance of standard DCell architecture with respect to throughput, delay and fault-tolerance by either modifying the architecture to support multiple shortest paths between any two servers, and/or implement a better fault-tolerant path selection algorithm for choosing the best path for routing the packets. For the two proposed topologies, the simulation results show an improvement in the throughput compared with the standard DCell, and the percentage of improvement increases as we increase n. The proposed topologies show an improvement in the throughput in the range of 3% to 5% (depending upon traffic model) for n= 16 as compared to DCell. This is mainly due to provisioning and utilization of multiple alternate paths between any pair of servers. In addition, the proposed topologies provide a graceful degradation in the performance in the presence of link or server failures.

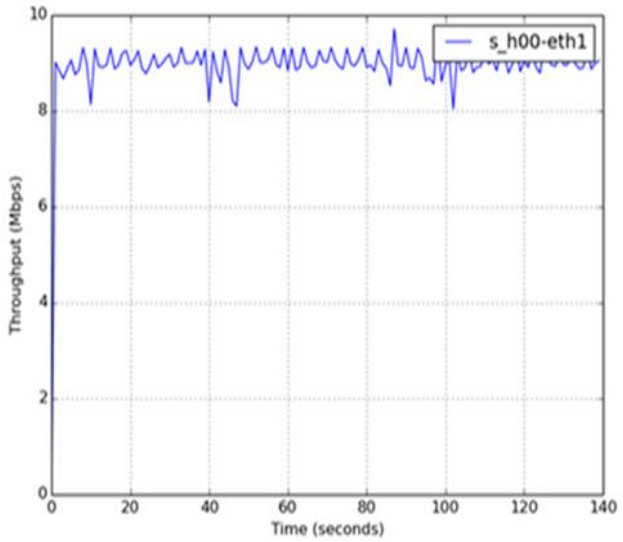

Fig. 9. Throughput under transient failures in the standard DCell topology (One-to-One traffic)

V. Conclusions

Due to the tremendous increase in the use of cloud-based services, the volume of data traffic to the data centers is on constant rise. Data center network (DCN) topologies provide a network within a datacenter connecting a huge number (~10,000) of servers with several switches and other networking devices. The most important features that a network topology must have are scalability, fault-tolerance, and QoS support for the desired applications. The provision of fault-tolerance is becoming even more critical as the number of servers and devices increase so does the probability of failures. Therefore, there is a strong need to have the support for fault-tolerance in the DCN architecture.

In this paper, we propose two new DCN topologies, called DCell-Star and DCell-Ring. The proposed topologies modify slightly standard DCell architecture. The modifications add a set of alternate shortest paths between any two servers in the topologies. The simulation results show that the proposed topologies provide higher throughputs and lesser latencies as compared to the standard DCell.. The improvement in the throughput ranges from 3% to 5% (depending upon the traffic model used) for n= 16, where n is the number of servers in basic level-0 of the topology. For large values of n, one can expect even higher percentage of improvements in throughput. Furthermore, as compared to the standard DCell, the proposed topologies have lesser diameters and lesser average shortest path lengths which makes it superior for using it in a large network with a large value of n. Reduction in the diameter metrics, especially in DCell-Star, results in low-latency for routing packets, even with large number of servers. In addition, the proposed topologies offer graceful performance degradation in case of a link or server failure.

Acknowledgment

The authors would like to thank American University of Sharjah, UAE, for providing a research grant (FRG16-R-13) in the support of the research work presented in this paper.

References

[1] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, p. 63, 2008.

[2] R. N. Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: A scalable fault-tolerant layer2 data center network fabric," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, p. 39, 2009.

[3] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: A scalable and fault-tolerant network structure for data centers," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 4, p. 75, 2008.

[4] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: A high performance, server-centric network architecture for modular data centers," *ACM SIGCOMM Computer Communication Review*, vol. 39, no. 4, p. 63, 2009.

[5] G. Wang, D. Andersen, M. Kaminsky, K. Papagiannaki, T. Ng, M. Kozuch, and M. Ryan, "C-Through: Part-time optics in data centers," *ACM SIGCOMM Computer Communication Review*, vol. 40, no. 4, p. 327, 2010.

[6] M. Bari, R. Boutaba, R. Esteves, L. Granville, M. Podlesny, M. Rabbani, Q. Zhang, and M. Zhani, "Data Center Network Virtualization: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 2, pp. 909-928, 2013.

[7] L. Brian, M. Aman, and T. Niharika. A Survey and Evaluation of Data Center Network Topologies. [Online]. Available: https://arxiv.org/abs/1605.01701, 2016.

[8] K. Bilal, S. U. Khan, J. Kolodziej, L. Zhang, K. Hayat, S. A. Madani, N. Min-Allah, L. Wang, and D. Chen, "A comparative study of data center network architectures," in *Proc. European Conference on Modeling and Simulation (ECMS)*, 2012, pp. 526-532.

[9] Y. Liu, D. Lin, J. Muppala, and M. Hamdi, "A study of fault-tolerance characteristics of data center networks," in *Proc. 42nd International Conference on Dependable Systems and Networks Workshops (DSN-W)*, 2012, pp. 1-6.

[10] Y. Liu and J. Muppala, "Fault-tolerance characteristics of data center network topologies using fault regions," in *Proc. 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks*, Budapest, Hungary, 2013.

[11] F. Yao, J. Wu, G. Venkataramani, and S. Subramaniam, "A comparative analysis of data center network architectures," in *Proc. IEEE International Conference on Communications*, 2014.

[12] M. Team. (2015). Mininet: An Instant Virtual Network on your Laptop (or other PC) – Mininet. Mininet.org. [Online]. Available: http://mininet.org/

[13] Y. Sun, J. Chen, Q. Liu, and W. Fang, "Diamond: An improved fat-tree architecture for large-scale data centers," *Journal of Communications*, vol. 9, no. 1, pp. 91-98, 2014.

[14] Z. Li, Z. Guo, and Y. Yang, "BCCC: An expandable network for data centers," *IEEE/ACM Transactions on Networking*, vol. 24, no. 6, pp. 3740-3755, 2016.

[15] T. Chen, X. Gao, and G. Chen, "The features, hardware, and architectures of data center networks: A survey," *Journal of Parallel and Distributed Computing*, vol. 96, pp. 45–74, 2016.

[16] J. Touch and R. Perlman. (2009). Transparent Interconnection of Lots of Links (TRILL): Problem and Applicability Statement. [Online]. no. 5556. IETF. Available: http://www.ietf.org/

[17] J. Pandey and R. Subramanian, "DCell: A scalable and fault-tolerant network structure for data centers," *Reproducing Network Research*, 2012.

[18] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang and A. Vahdat, "Hedera: Dynamic flow scheduling for data center networks," *NSDI*, vol. 10, pp. 19-19, 2010.

[19] POX Controller Tutorial | SDN Hub. (2011). [Online]. Available:http://sdnhub.org/tutorials/pox/.

[20] OpenFlow Tutorial-OpenFlow Wiki. (2011). [Online]. Available:http://archive.openflow.org/wk/index.php/OpenFlow_Tutorial.

[21] Router Ryubook 1.0 documentation. (2014). [Online]. Available:https://osrg.github.io/ryu-book/en/html/rest_router.html#setting-static-routes-and-the-default-route

[22] V. Guent. iPerf - The TCP, UDP and SCTP network bandwidth measurement tool. (2013). [Online]. Available: https://iperf.fr/

[23] T. Fisher. How to Use the Ping Command in Windows. Lifewire. (2016). [Online]. Available: https://www.lifewire.com/ping-command-2618099.

Rana E. Ahmed received his PhD degree in Electrical Engineering from Duke University, NC, USA, in 1991. He is currently working as an Associate Professor at the Dept. of Computer Science and Engineering, American University of Sharjah, United Arab Emirates. His current research interests include wireless networks, fault-tolerant computing, datacenter architectures, software engineering, and architectures for signal processing applications. He has published 60 research papers in international journals and conferences, and has authored three book chapters.

Heba Helal received her MSc degree in Computer Engineering from the American University of Sharjah, United Arab Emirates. Her research interests are in the areas of computer networking and software engineering.