**Simulation of DCN Topologies using ns-3**

Suraj Ketan Samal, Upasana Nayak

University of Nebraska-Lincoln

May 5, 2015

*ssamal@cse.unl.edu* , [*unayak@cse.unl.edu*](mailto:unayak@cse.unl.edu)

1. **Introduction to Problem:**

With internet playing a significant role in life-cycle of every individual, and recent advancements like multimedia streaming, social networking, scientific big-data and need for high-speed internet, the amount of data handled by data-centers is increasing at a very rapid rate. As per Cisco global cloud Index forecast [1], the global data-center traffic is increasing at 23% per year and will reach 8.6 *zettabytes* by the end of 2018. Hence, the need to manage the data-center traffic efficiently has become extremely inevitable. However, traditional data-centers are unable to handle this growth of traffic and face challenges including scalability, high-bisection bandwidth, reduced latency, cost, fault-tolerance and agility [2]. Use of artificial intelligence and machine-learning techniques for design and monitoring, greener and power efficient data-centers and optical networking capabilities are some of the recent innovations being explored to address the problem. One of the important aspect affecting most of the above challenges is the interconnection topology that connects the various components (e.g switches, servers and other network devices) [3] of a data-center. A good amount of research has currently focused on building efficient and intelligent topologies.

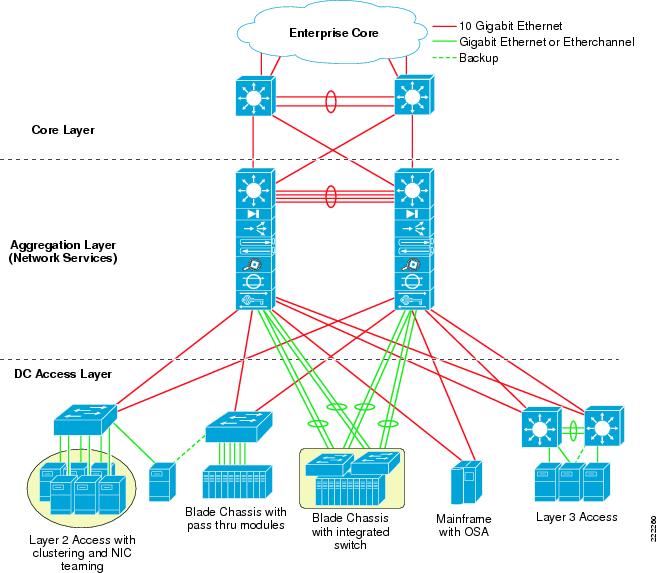
In this project, we plan to study and simulate various important data-center topologies proposed in literature (*Fat-Tree* and *Bcube*) using *ns-3* simulator. We also simulate the (*HyScale* [4]) topology, earlier proposed by our team with similar traffic conditions and compare them with the simulated topologies.

1. **Updated Scope of the Project:**

Because of the amount of efforts spent in set-up and learning the basics of *ns-3* simulator, we had to reduce the scope of our project from the initial proposal with the following changes:

1. The original scope included simulation of *six* topologies namely fat-tree, flattened butterfly, Bcube, DCell, FiConn and HyScale I and HyScale II. We had to modify the scope to include only three topologies now – (a) Fat-Tree, (b) BCube and (c) HyScale.
2. The original scope included comparison among the topologies along five parameters namely Bisection Bandwidth, Latency,Fault tolerance, cost and throughput. We have reduced the scope to include only two comparison parameters – (a) latency and (b) throughout.
3. **Introduction to DCN:**

Data Center Networks (DCN) consist of a pool of interconnected network elements that serve the entire internet. Traditional data-centers usually were constructed in a layered architecture as shown in *Figure-1*.



*Figure-1: Cisco Data Center Architecture, Retrieved from* [*http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data\_Center/vmware/vmware/VMware.html*](http://www.cisco.com/c/en/us/td/docs/solutions/Enterprise/Data_Center/vmware/vmware/VMware.html)*.*

Each layer consists of network elements (switches, servers etc), that enables connectivity to other layers. At the top is the *core-layer* which is connected to the internet using high speed data links to serve to-and fro traffic from the internet. At the middle consists of the *aggregation layer* which enables traffic communication between the *core* and *access* layer. Some redundancy is typically built in to enable multiple paths to servers in case of disaster recovery scenarios. Access layer typically include switches and the actual servers that serve data to the outside world.

Modern datacenters employ various interconnection topologies for high efficiency. Modern datacenter architectures can be divided into two major categories – *switch centric* and *server-centric*. Switch centric architectures typically have all routing process performed by the intermediate network elements (eg. switches) and have servers only at the end-points. Server centric architectures include the servers also in assisting with the routing process. Some important architecture includes Fat-Tree, BCube, DCell and FiConn. For a description and quantitative comparison of important data-centre topologies, the reader can refer to [2].

1. **Major Challenges faced by Modern DCN:**

In this section, we briefly discuss some of the important challenges faced in modern datacenter networks (DCN).

1. *Scalability*: With a rapid growth in the data transfer forecasted in future, it is highly important that the network devices and links support very high numbers of servers and data-communication between them. Additionally, addition/removal of new elements in real-time should also be supported
2. *Bisection bandwidth:* Bisection bandwidth refers to the maximum bandwidth the network can support if the network is segmented to two equal parts. Traffic patterns observed recently in DCNs indicate that there is a huge amount of traffic being sent within the servers within close locality due to caching and other reasons. It is an important measure of network performance originally used in VLSI circuit design.
3. *Latency:* With high-speed links, the time taken by a packet to reach from a source to a destination is no longer limited by the transmission delays of the link but by the properties of switches and other network elements (like buffers, queuing algorithm used etc). Design of large DCN networks should ensure that the average latency is low and doesn’t fall below a particular threshold
4. *Fault Tolerance*: Another important challenge in DCN is to be able to design highly reliable networks with very low packet drops using various capabilities like building redundant content, multiple paths and building auto-fault detection mechanisms.
5. *Cost:*The cost of entire infrastructure required to build and maintain the data-center is a critical challenge with high demand of data-transfer. This cost involves cost servers, switches, ports and cables and resources required to maintain the data-center servers.
6. **Important DCN Topologies:**

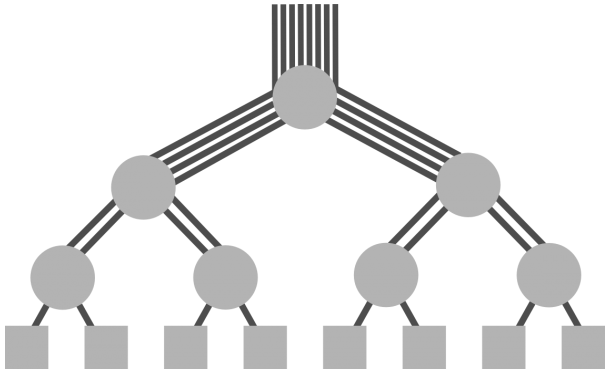
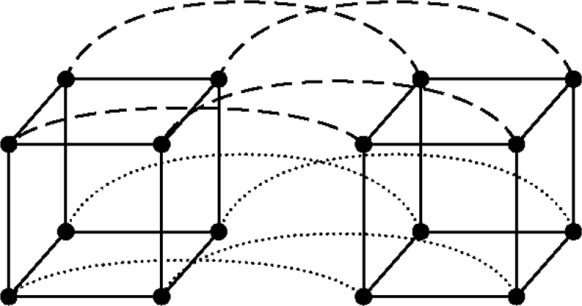
Here we briefly introduce the three important topologies we evaluated as part of the project.

1. *Fat-Tree :*

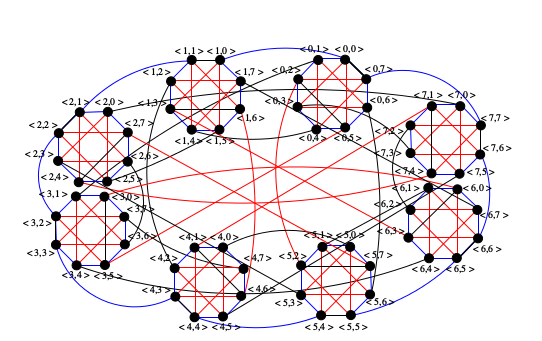
This topology was initially designed by Charles E. Leiserson, MIT in 1985((Figure 2a). It involves nodes connected in a complete binary-tree fashion with servers being the leaves of the binary-tree and the root acting as the interface to the network. The links at each level are designed in a way that each server gets a dedicated bandwidth/path to every other server. The links become fatter as we go from leaves to the root. This topology is considered one of the fundamental ones in network design, inter-processor networks and other areas.

1. *BCube :*

*BCube* is another classic server centric data-center topology by Microsoft Group, Asia in 2009(Figure 2b). It is an implementation of *BCube* suggested in Mathematics which is recursive consists of connecting two BCubes of smaller order and connecting isomorphic nodes with a separate link. A number of servers can then be connected at any node.

** **

*(a): Fat-Tree Topology. (b) BCube Topology (k=2).*

****

*Figure-2 (a): Fat-Tree Topology. (b) BCube Topology, (c) HyScale Topology (T=8)*

1. *HyScale :*

*HyScale* topology was proposed in 2012. It is a recursive topology possessing necessary graph-theoretic properties necessary for enhancing throughput and reliability in large scale DCN networks (Figure 2c). It falls under the category of hybrid data-center architectures where switches have are connected in a recursive fashion through various levels. Each switch/node possesses a *hyscale* address and can support a number of servers connected to it.

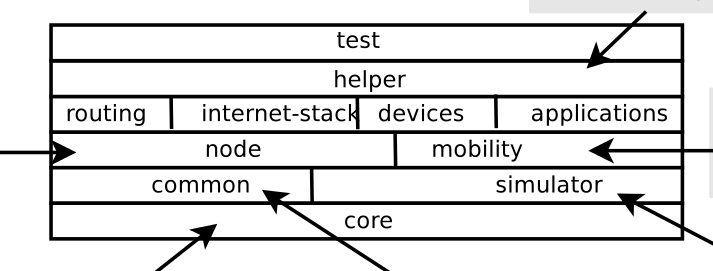
Some important theoretic properties of the three topologies are shown in Table 1:

|  |  |  |
| --- | --- | --- |
|  | ***Servers*** | ***Diameter*** |
| *FatTree* | */4*  *(n= number of groups/pods)* | *k+1* |
| *BCube* | *(n = number of servers connected in basic unit,*  *k= recursion depth)* | *4* |
| *HyScale* | *(a = number of servers connected at each node, k= recursion depth, T = number of nodes in base model)* | *4k+2* |

*Table 1 Some important theoretic properties of the three FatTree, BCube and HyScale*

1. **Overview of Implementation Tool (ns-3):**

*ns-3* is one of the popular open-source tools can be used to simulate network systems. It is a discrete event simulator primarily for research and educational use. It is a complete revamp of *ns-2* which is used extensively for simulation in research and industry. *ns-2* is designed in *C++* at its core, *Tcl/oTcl* at its frontend and supports a variety of network technologies and architectures. In comparison, *ns-3* is completely designed in *C++* with aiming to support a wide range of advanced features like exporting to standard xml outputs and *gnuPlots*, packet-capture capabilities at supported interfaces and supporting multiple visualizers (*netAnim* and *pyViz*). *ns-3* still carries the same conceptual design as *ns-2* however doesn’t have as extensive capabilities as *ns-2*. A diagram of various layers present in *ns-3* architecture can be seen in *Figure 3*.



*Figure 3: Various layers of ns-3 architecture (Source: ns-3 manual [6])*

1. **Other Tools Explored**

We explored another tool *netkit* [9] which is open-source and claimed to be simple to perform experiments with low cost and little effort. *Netkit* is a network emulator that uses light-weight virtual machines to support features that seem to be real. *Netkit* seemed to work well for smaller number of servers which needed to be configured manually, but didn’t seem practical for testing *DCN* topologies with high number of servers.

1. **Implementation Details:**

We simulated the three topologies described in *Section 5* using *ns-3* (*v3.22*) tool. Some important parameters of our setup in all three topologies are shown in Table 2 below.

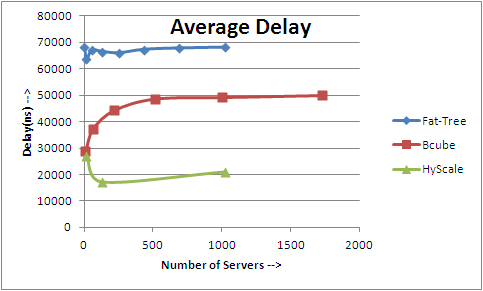
|  |  |  |  |
| --- | --- | --- | --- |
| **Parameters** | **Fat-Tree** | **BCube** | **HyScale** |
| *Simulation Time* | 100sec | 100sec | 100sec |
| *PacketSize* | 1024bytes | 1024bytes | 1024bytes |
| *DataRates between switches* | 1Mbps | 1Mbps | 1Mbps |
| *DataRates between switches and hosts* | 1000Mbps | 1000Mbps | 1000Mbps |
| *Communication Pattern* | Random selection of two hosts and sending data between them | Random selection of two hosts and sending data between them | Random selection of two hosts and sending data between them |
| *Traffic Flow Pattern* | Exponential Random | Exponential Random | Exponential Random |
| *Routing Protocol* | Nix-vector(ns-3) | Nix-vector(ns-3) | Nix-vector(ns-3) |
| *Animator* | NetAnim | NetAnim | NetAnim |
| *Variable parameter (refer to Table 1)* | *n varied (2-12)* | *k=2, n varied(2-12)* | *a=2, T=8, k varied from (1-3)* |
| *SourceFile* | Fat-Tree.cc | BCube.cc | HyScale.cc |
| *Output Statistics* | *statistics/Fat-Tree.xml* | *statistics/BCube.xml* | *statistics/HyScale.xml* |
| *Output Files* | *statistics/fat-tree-stats.csv* | *statistics/bcube-stats.csv* | *statistics/hyscale-stats.csv* |

*Table 2: Simulation Parameters used in our ns-3 implementation.*

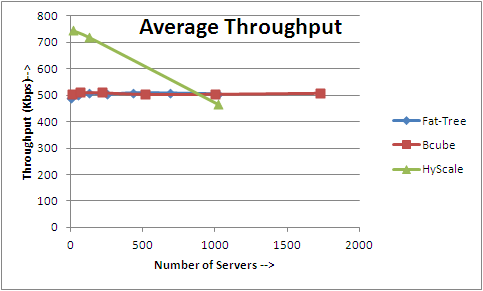
We extended some of the existing implementation of *Fat-Tree* and *BCube* topologies as mentioned in [5, 8] to be able to support *NetAnim* and export relevant statistics needed for our analysis. We developed a new implementation of *HyScale* using similar characteristics used for *Fat-Tree* and *BCube.*

1. **Results and Observation**

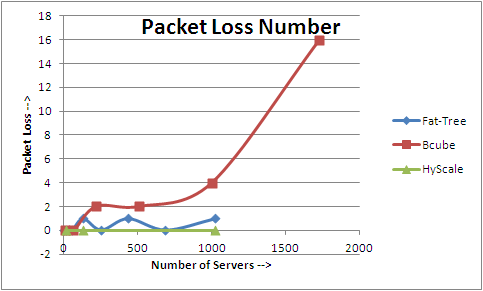
The following are some important results obtained from running the simulations on *ns-3* on a personal computer with Linux installed on it. The same can also be found in “*stats.xls”* as part of this submission. Due to the limitation of time and processing capability of the computer, we couldn’t simulate for very high number of servers, but believe that the same pattern should take over in such cases.



*Figure-5: Observed average delay with exponential random traffic pattern.*



*Figure-6: Observed average throughput with exponential random traffic pattern.*

**

*Figure-7: Observed packet loss with exponential random traffic pattern.*

***Important Observations:***

* As expected, the average delay (Figure 5) observed in Fat-tree topology is very high due to the longer average distance and only single path between any pair of nodes. In recursive topologies, the average path length between any pair of nodes is very small and hence a lower delay. However *HyScale* (~25ms) performed about twice as better than *BCube* (~50ms)which is most likely because of additional redundancy and symmetric properties. Additionally, in *BCube* the average delay increases at a greater speed with number of servers where as with *HyScale* the growth seems to be fairly slow.
* The average throughput seems to be fairly constant with *BCube* and *Fat-Tree* (~500kbps) where as with *HyScale* it seems to decrease with increase in number of servers. *HyScale* seems to perform better with lower number of servers (~700Kbps). We donot seem to have a concrete reason for such behavior and intend to investigate this in future work.
* The packet loss observed with *BCube*(~16)is high and increases with number of servers. With *FatTree(0-1),* the loss of packet oscillates with very small number. This seems to be an expected behavior as with *FatTree* every pair of nodes have a dedicated path and hence paths would be less congested. *HyScale* seems to perform best with no packet-loss observed.

1. **Future Work**

* Simulate other DCN topologies (FiConn, DCell and HyScale II) and compare with each other.
* Simulate the DCN topologies for varying types of traffic types observed in DCN.
* Explore and compare fault-tolerance and bisection-bandwidth parameters of various topologies.
* Support and Simulate DCN topologies for optical network switching (OBS and OCS).

1. **References:**

[1] Cisco Global Cloud Index: Forecast and Methodology, 2013–2018. Retrieved from

[https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud\_Index\_White\_Paper.pdf](https://www.cisco.com/c/en/us/solutions/collateral/service-provider/global-cloud-index-gci/Cloud_Index_White_Paper.pdf%20)

[2] Bilal, Kashif, et al. "Quantitative comparisons of the state of the art data center architectures." Concurrency and Computation: Practice and Experience 25.12 (2013): 1771-1783.

[3] Bilal, Kashif, Samee Ullah Khan, and Albert Y. Zomaya. "Green Data Center Networks: Challenges and Opportunities." Frontiers of Information Technology (FIT), 2013 11th International Conference on. IEEE, 2013

[4] Saha, S., Deogun, J. S., & Xu, L. (2012, June). Hyscale: A hybrid optical network based scalable, switch-centric architecture for data centers. InCommunications (ICC), 2012 IEEE International Conference on (pp. 2934-2938). IEEE.

[5] D. Wong, K.T. Seow, C.H. Foh and R. Kanagavelu, “Towards Reproducible Performance Studies of Datacenter Network Architectures Using An Open-Source Simulation Approach”, Proceedings of the IEEE Global Communications Conference (GLOBECOM’13), December 2013, Atlanta, GA, USA.

[6] NS3-manual. At <https://www.nsnam.org/docs/release/3.11/manual/ns-3-manual.pdf>

[7] NS3-Tutorial. At <https://www.nsnam.org/docs/release/3.22/tutorial/singlehtml/>

[8] NTU-DSI-DCN Project. At <http://ntu-dsi-dcn.github.io/ntu-dsi-dcn/>

[9] Poor man’s tool to support Computer Networking. At [www.netkit.org](http://www.netkit.org)

**----------------------------------------------------**