A Software Defined Network Scheme for Intra Datacenter Network Based on Fat-Tree Topology

Siquan Hu$^{1,2*}$, Xinyu Wang$^1$ and Zhiguo Shi$^{1,3}$

1 School of Computer and Communication Engineering, University of Science and Technology Beijing, Beijing, China
2 Shunde Graduate School, University of Science and Technology Beijing, Foshan, China
3 Beijing Big Data Center, Beijing, China
* Email: husiquan@ustb.edu.cn

Abstract. Nowadays datacenters face a big challenge, because the development of cloud services requests a large amount of capital input, high maintenance skills and cost. The major disadvantages of current datacenter network include the high cost of equipment and maintenance, QoS, failure recovery time, network virtualization etc. In this paper, we proposed a software defined datacenter network. The programmable controller is responsible for the management of the traffic scheduling including topology discovery, routing and Quality of Service. The SDN is based on Fat-tree network architecture and max-min fairness. A Genetic Algorithm optimized Radial Basis Function neural network is used to compute the service weight for maximizing the utilization of network resource. An experiment of the network bearing different kinds of services based on OpenFlow is described. This well-designed fat-tree network has high efficiency in traffic distribution, and has integrated the traditional network architecture and new SDN technology.

Keywords. Fat-Tree; software defined network; datacenter network.

1. Introduction
SDN is a new network architecture invented by Stanford University research team. Its core concept is the separation of control plane and data plane [1]. The biggest characteristic of SDN is the fusion of network and service, which allows network to respond to business more rapidly and flexibly [2]. In fact, service control over network is relatively weak in traditional network.

Unlike inter datacenter network which has a relative low latency sensitivity, intra datacenter network has a very stringent requirement of latency and accuracy [3]. In this paper, we present a SDN based intra datacenter network. It is a new architecture leveraging software controller in datacenter network within a fat-tree topology.

2. Related Work
Since firstly introduced in the popular Thinking Machines CM-5 in 1991, fat-tree networks have become popular during the past few years [4]. They have negligible difference in terms of performance on full bisection bandwidth of fat-tree network which is considered under-utilized for HPC (high performance computation) [5]. Current algorithms can provide different QoS to services with different demands, because those algorithms are based on source and destination address without consideration about service demand [6]. For these reasons, we proposed SDNFT, an SDN based on fat-tree[7].
SDNFT can provide low-cost traffic scheduling in regular situation and software-based adjustment when deterministic routing cannot satisfy the demand of transaction with high priority [8]. The design of network architecture is important because the traffic of intra-datacenter network traffic requires a significant bandwidth due to an aggregate of over thousands of computers [9]. There are several network schemes aiming at fulfilling the service demand being put forward such as fat-tree [10], VL2 etc [11]. Since fat-tree can provide relatively good performance and is being widely adopted, we propose a SDN based on fat-tree structure [12].

3. Architecture

3.1. Overview
Figure 1 shows the basic architecture of SDNFT. A centralized software-based controller is on the top of it, providing complete visibility and control over the network. We used typical 2-level fat-tree and D-mod-k to simplify our implementation. We define two conditions in the network design.

**Condition A** represents the situation when deterministic routing algorithm can totally balance the network traffic without congestion. This condition occurs mostly at the beginning of the traffic or traffic volume is relative low and steady.

**Condition B** represents the situation when deterministic routing algorithm cannot fulfill the demand of high priority service. This scenario appears when large flows of high priority service burst and network is congested.

3.2. Switch
Our design deploys Openflow switches. Figure 2 shows the main components of the switch including flow tables, an Openflow Channel and a group table. The sequentially numbered flow tables in the switch constitute an Openflow pipeline which defines how packets interact with the flow entries in the flow tables. Incoming packet is first matched against flow entries of flow table 0 and the outcome decides if other flow tables would be used. The specially configured parts are underlined in table 1.

![Figure 1. Architecture of SDNFT.](image1)

![Figure 2. OF switch.](image2)

| Match Fields | Priority | Counters | Instructions | Timeouts | Cookie |
|--------------|----------|----------|--------------|----------|--------|

4. Routing
The routing algorithm has two parts according to the following two conditions.
4.1. D-mod-k for Condition A
In condition A, deterministic routing algorithm can satisfy the demand of network traffic and no congestion takes place. We chose D-mod-k for the deterministic algorithm. Its core concept is to distribute the flows forwarded to consecutive destinations among different ascending links, reaching different switches in the next stage. Figure 3 shows an example of the destination distribution in our fat-tree based network architecture. The packets will be forwarded from the source node to a nearest common ancestor (NCA) of the SD (source-destination) pair, then downwards to the destination node.

The distribution scheme will be computed by controller within the DR module.

In condition A, the network traffic is relative low and steady. So, the D-mod-k algorithm, which is easy and fast computing, is capable to control the relative simple condition.

4.2. Max-min Fairness for Condition B
(1) Launch condition
Once total demand D of flows forwarded to the output port K on switch S is larger than a threshold, i.e. 1.2 times bandwidth of K, a packet-in message will be sent to controller as a congestion alarm. We assume that switch S has two output ports, K and K’. Then the controller will send a packet-out message including request for flow stats to switch S, and find out the stats of flows being forwarded through port K and K’ in real time. The K-forwarding-service group and K’-forwarding-service group will be matched against to the service-priority table, and the weights of all services in K and K’ group will be evaluated. Only if, the weight of K group is larger than K’ group for a threshold, MMF module will be started. We set up this constriction to maximize the utilization of network resource to ensure the QoS of high priority services instead of rebalancing frequently caused by low priority services.

(2) Fair share
We associated a bandwidth function with every services specifying the bandwidth allocation to a service given the flow’s relative weight. We call this fair share(e.g., figure 4).

![Figure 3. D-mod-k.](image)

![Figure 4. Fair share for applications.](image)

(3) Simple Max-min fairness
To accommodate QoS and latency caused by bandwidth computation in overall network, we define a Simple Max-Min Fairness (SMMF) according to the characteristic of our network architecture. Based on this characteristic, our SMMF compute the network resource between switches of stage 1 and stage 2. The SMMF wouldn’t interrupt the traffic outside of this area, saving computation and improving the efficiency of algorithm. We used cost of an edge to represent its latency.

The basic steps are the following:
- Initialize the flow group with its most preferred tunnel.
- Allocate bandwidth by giving equal fair share to each App.
- When a link becomes bottleneck, freeze all tunnels containing this link.
- Iterate until every tunnel is frozen or all Apps are satisfied.
5. Experiment
To evaluate our proposed design, we built a simulation environment with Opendaylight controller and mininet. Its Model-Driven Service Abstraction Layer is set of infrastructure services aimed at providing common and generic support to develop application and plugin against one set of APIs. We implemented the DR and MMF modules as plugins.

We use three network services, service A, B, C to simulate the majority of a datacenter network service. Service A has the highest priority 10 with a constant bandwidth demand without any disruption. Service B has a priority 5, which is used by the researchers in the datacenter. Service C has a relatively low priority 3, which is used as the experimental data for students.

We simulated the practical data forwarding with the ODL controller and Mininet in the virtual environment. We emulated the fat-tree network shown in figure 1 with mininet and set the links with 10Mbps bandwidth. Service A and service B mentioned before, were defined with demand of 8Mbps, 6Mbps. Service C has an infinite demand. For simplifying our test, we set the source and destination of these services exactly according to the example proposed in 4.2. The result is as follows:

Table 2. Weight allocation.

|       | Specified priority | Latency sensitivity | Date volume | Weight |
|-------|--------------------|---------------------|-------------|--------|
| Service A | 10                 | 10                  | 10          | 10     |
| Service B | 5                  | 6                   | 8           | 6      |
| Service C | 2                  | 4                   | 6           | 4      |

Table 2 shows the weight allocation computed by GA-RBF. The parameters in the table are quantization of the real parameters according to the rule-10 is the highest while 0 is the lowest. The quantization are the relative values. As we see, service A with the highest parameters in all three reference standards get the highest weight. Service B and C get weight 6 and 4 respectively. The bandwidth allocation compared to the desired allocation is shown in figure 5.

![Figure 5. Result of bandwidth allocation.](image)

As figure 5 shows, Service A is allocated 5Mbps and 3Mbps from tunnel 1 and tunnel 2 respectively. Service B is allocated 3Mbps from tunnel 1 and tunnel 2 respectively. Service C is allocated 2Mbps and 4Mbps from tunnel 1 and tunnel 2 respectively. All three services are satisfied without congestion.

6. Conclusion
We proposed SDNFT, which is a novel approach to embed SDN concept into traditional network design. With the basic well designed fat-tree network which gets high efficiency in traffic distribution, we add the software based control function into the relative static deterministic data forwarding algorithm. We integrate traditional well designed network architecture and new SDN technology. Our design has advantages including fast convergence in regular condition and dynamic load balance based on max-min fairness. DR module can control the network with a low cost during the steady phase of
the traffic. SMMF module can provide dynamic load balance during the burst phase. The GA-RBF algorithm can provide QoS due to different service demand and priority.

Acknowledgments
This work is supported by the National Natural Science Foundation of China (Grant No. 61977005), the Scientific and Technological Innovation Foundation of Shunde Graduate School, USTB (BK19CF003) and Beijing Municipal Science and Technology Project (No. Z201100004220010).

References
[1] Chica J, Imbachi J C and Botero J F 2020 Security in SDN: A comprehensive survey Journal of Network and Computer Applications 159 102595.
[2] Guck J W, Bemten A V, Reisslein M, et al. 2017 Unicast QoS routing algorithms for SDN: A comprehensive survey and performance evaluation IEEE Communications Surveys & Tutorials (99) 1-1.
[3] Dehury C K and Sahoo P K 2020 Failure aware semi-centralized virtual network embedding in cloud computing fat-tree data center networks IEEE Transactions on Cloud Computing (99) 1-1.
[4] Mohtavipour S M, Mollajafari M and Naseri A 2020 A novel packet exchanging strategy for preventing HoL-blocking in fat-trees Cluster Computing 23 (7).
[5] Kadim U N and Mohammed I J 2020 SDN-RA: An optimized reschedule algorithm of sdn load balancer for data center networks based on QoS IOP Conference Series: Materials Science and Engineering 928 (3) 032057.
[6] Rg A, Es A, Pjg A, et al. 2021 Towards an efficient combination of adaptive routing and queuing schemes in Fat-Tree topologies Journal of Parallel and Distributed Computing 147 46-63.
[7] Fujiki D, Ishii K, Fujiwara I, et al. 2017 High-bandwidth low-latency approximate interconnection Networks IEEE International Symposium on High Performance Computer Architecture (IEEE).
[8] Poovendran R and Sumathi S 2018 An low-power area-efficient routing analysis and optimization for a fat-tree-based optical network-on-chip (ONoC) Architectures Sensor Letters 16 (11) 862-876.
[9] Gliksberg J, Capra A, Louvet A, et al. 2019 High-quality fault-resiliency in fat-tree networks (Extended Abstract) 2019 IEEE Symposium on High-Performance Interconnects (HOTI) (IEEE).
[10] Sun L, Yong-Yi R, Bo-Wen Y, et al. 2017 Dynamic switching mechanism for multi-tree multicast in SDN-based fat-tree data center network Journal of Chinese Computer Systems 038 (004) 761-765.
[11] Hauser F, Schmidt M, Haberle M, et al. 2020 P4-MACsec: Dynamic topology monitoring and data layer protection with MACsec in P4-based SDN IEEE Access (99) 1-1.
[12] Nallusamy P, Saravanen S and Krishnan M 2020 Decision tree-based entries reduction scheme using multi-match attributes to prevent flow table overflow in SDN environment International Journal of Network Management 031 (004) e2141.