國 立 交 通 大 學

電信工程研究所

碩 士 論 文

雲端資料中心之動態且互斥之
資料鏈結層繞徑研究

Dynamic and Disjoint Layer-2 Routing in Cloud Datacenters

研究生：劉耕含

指導教授：王蒞君

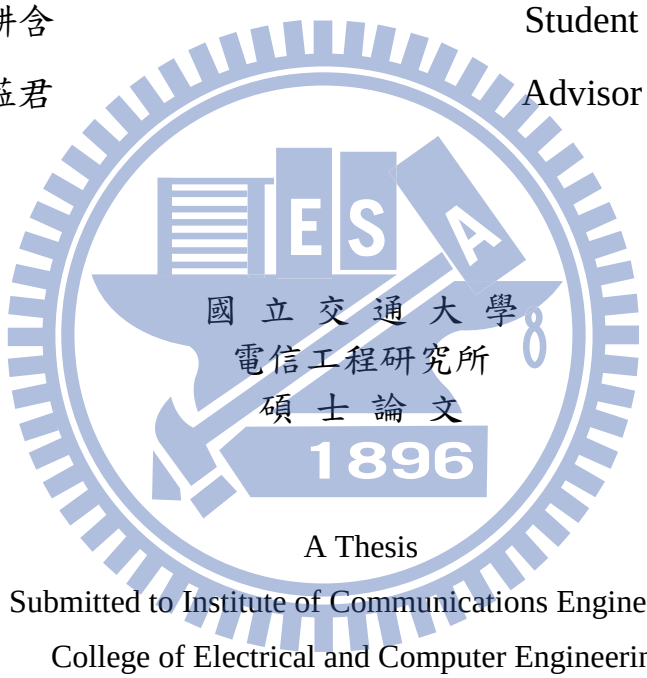
中 華 民 國　　一百零一 年 六 月

雲端資料中心之動態且互斥之資料鏈結層繞徑研究

Dynamic and Disjoint Layer-2 Routing in Cloud Datacenters

研 究 生：劉耕含　　Student：Gen-Hen Liu

指導教授：王蒞君　　Advisor：Li-Chun Wang



A Thesis

Submitted to Institute of Communications Engineering

College of Electrical and Computer Engineering

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master of Science

In

Communication Engineering

June 2012

Hsinchu, Taiwan, Republic of China

中華民國 一百零一 年 六 月

Dynamic and Disjoint Layer-2 Routing in Cloud Datacenters

A THESIS Presented to

The Academic Faculty By

Gen-Hen Liu

In Partial Fulfillment

of the Requirements for the Degree of

Master in Communications Engineering

Institute of Communications Engineering

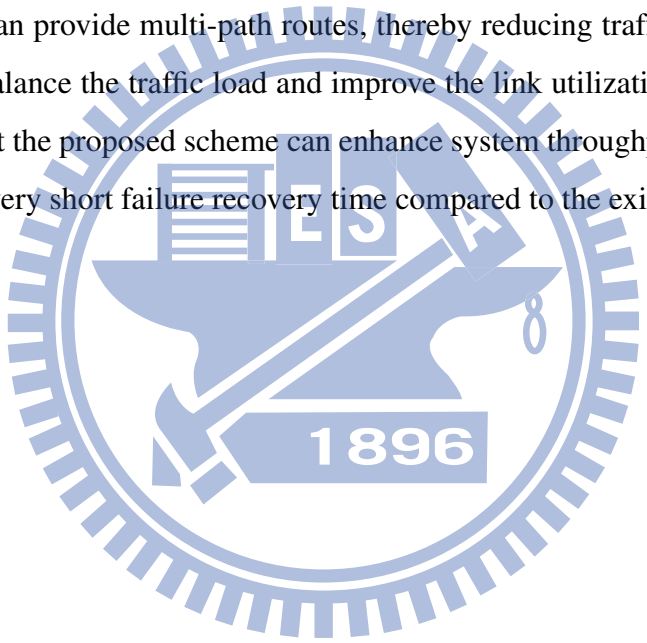
College of Electrical and Computer Engineering

National Chiao-Tung University

June, 2012

Abstract

This thesis presents an improved layer-2 routing algorithm, called dynamic and disjoint edge node divided spanning tree (D^2ENDIST), to overcome the issues of the single path route and unbalanced link utilization in cloud datacenters. D^2ENDIST consists of two key schemes: *(1) disjoint ENDIST routing* and *(2) reroute by dynamic reweights*. The former scheme can provide multi-path routes, thereby reducing traffic congestion. The latter scheme can balance the traffic load and improve the link utilization. Our experimental results show that the proposed scheme can enhance system throughput by 25% subject to the constraint of very short failure recovery time compared to the existing ENDIST scheme.

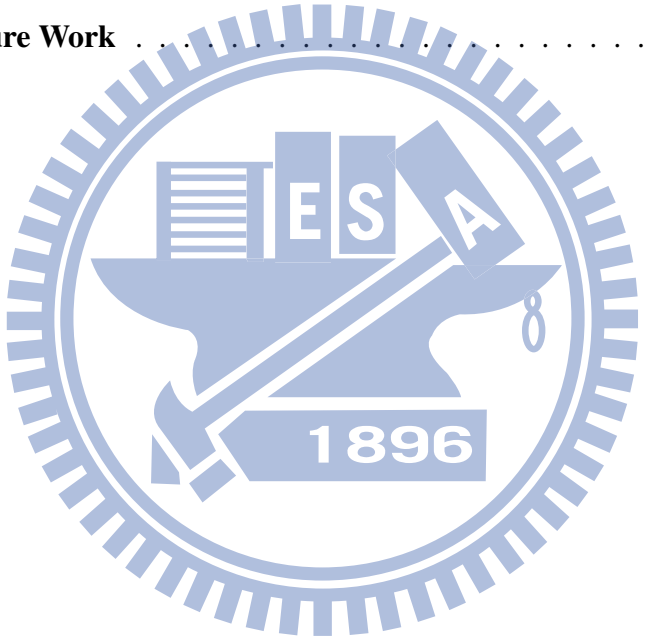


Acknowledgments

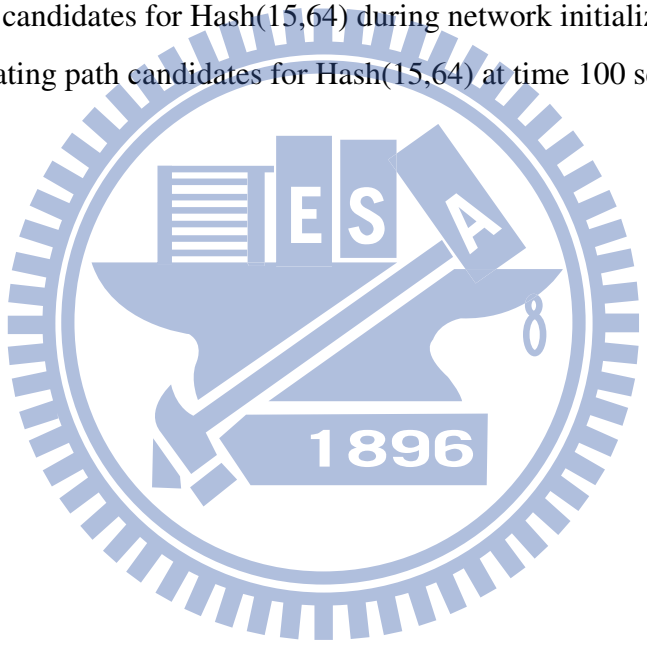
I would like to thank my parents and older sisters. They always give me endless supports. I especially thank Professor Li-Chun Wang and Charles H.-P. Wen who gave me many valuable suggestions in my research during these two years. I would not finish this work without his guidance and comments.

In addition, I am deeply grateful to my laboratory mates, Ang-Hsun, I-Cheng, Tsung-Chan, Yen-Ming, Chen-Hsiao, Wen-Pin and junior laboratory mates at Mobile Communication and Cloud Computing Laboratory at the Graduate Institute of Communications Engineering in National Chiao-Tung University. They provide me much assistance and share much happiness with me.

Contents



List of Tables



List of Figures


E S A
1896

CHAPTER 1

Introduction

Cloud computing is a hot research topic in recent years because the computation and data transmitting increase dramatically. Since datacenters have powerful computation resource, the networks must support sufficient bandwidth to maintain the service. Hence, how to keep in good networks condition is an important issue. In this thesis, we discuss the critical issues in datacenter networks such as routing and networking topology. Finally, we propose our solution and show the advantage of our method by series of simulation results.

1.1 Motivations

Cloud computing coordinates a lot of collocated servers to provide versatile applications. The issues of networking collocated servers are different from those in the current wide area networks (WAN). Thus, many cloud datacenters develop new layer-2 routing topologies, while WAN considers the layer-3 routing . Basically, the layer-2 routing can have faster packet forwarding speed compared to the layer-3 routing topology because of lower overhead. However, the layer-2 routing schemes may not be scalable when the number of servers increases.

Many newly proposed topologies for cloud datacenters emphasize on the scalability issue, including VL2 [1], PortLand [2], and BCube [3], etc. VL2 can achieve low cost in high-speed hardware implementations, but its Equal Cost Multi-Path (ECMP) scheme faces the problem of arriving packets with incorrect orders. PortLand aims at supporting a "plug-and-play" large-scale datacenter, but it does not have the multi-path property and may face the serious congestion problems.

Fig. 1.1 illustrates the motivation of this work. In a fat-tree topology with 10 nodes, the layer-2 spanning tree protocol (STP) and shortest path bridge (SPB) protocols may frequently choose the same specific path. In the example when nodes 6 and 7 try to connect to nodes 8 and 9, STP protocol result in paths 6-2-0-4-8 and 7-2-0-4-9, respectively, and a congestion occurs at node 2. One can also observe that nodes 1, 3, and 5 are not used. As shown in Fig. 1.2, one can easily select a better path for the request of connecting node 6 to node 8, and node 7 to node 9. Specifically, paths 6-2-0-4-8 and 7-3-1-5-9 can avoid congestion and balance traffic load as shown in Fig. 1.2. This observation motivates us to propose a routing algorithm that can achieve both congestion avoidance and load balance on the fat-tree network topology.

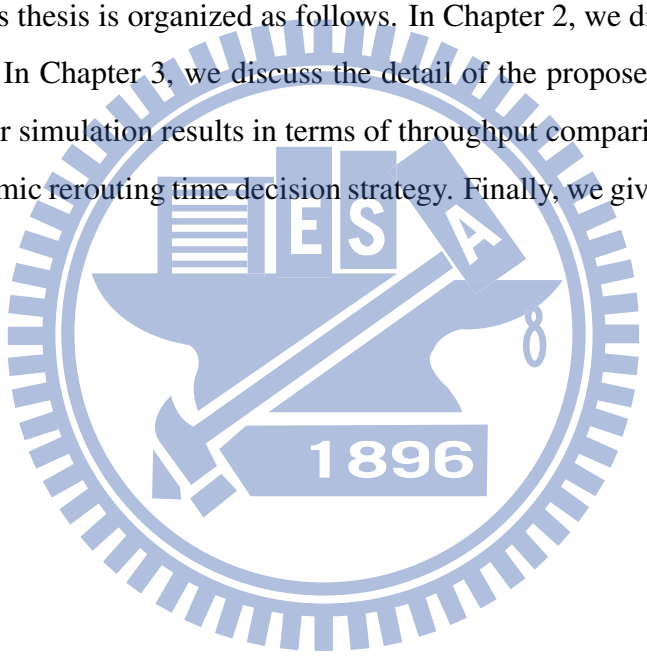
1.2 Problem and Solution

In datacenter networks, how to fully utilize the networking resource to avoid system bottleneck is a crucial issue. Since the request to datacenter grows daily, the amount of computation and data transmitting become huge. For this reason, it will easily cause the system bottleneck if we don't have adaptive strategy. Hence, we propose dynamic and disjoint edge node divided spanning tree (D^2ENDIST), consisting of two techniques (1) *disjoint ENDIST routing* to provide multipaths and (2) *reroute by dynamic reweights* to react the network status. Our experimental results show that the proposed scheme can enhance system throughput by 25% and achieve very short failure recovery time compared to the ex-

isting ENDIST scheme [4]. Also, D^{2}ENDIST achieves 10% throughput improvement than second best Hash-based routing (HBR) [5] under TCP traffic model. To summarize, the main contribution of this paper is to develop a multi-path load-balancing routing algorithm in a scalable large-scale network that can recover failure links without the issue of packet out of sequence.

1.3 Thesis Outline

The rest of this thesis is organized as follows. In Chapter 2, we discuss the related work in the literature. In Chapter 3, we discuss the detail of the proposed techniques. In Chapter 4, we show our simulation results in terms of throughput comparison, link failure recovery time and dynamic rerouting time decision strategy. Finally, we give our concluding remarks in Chapter 5.



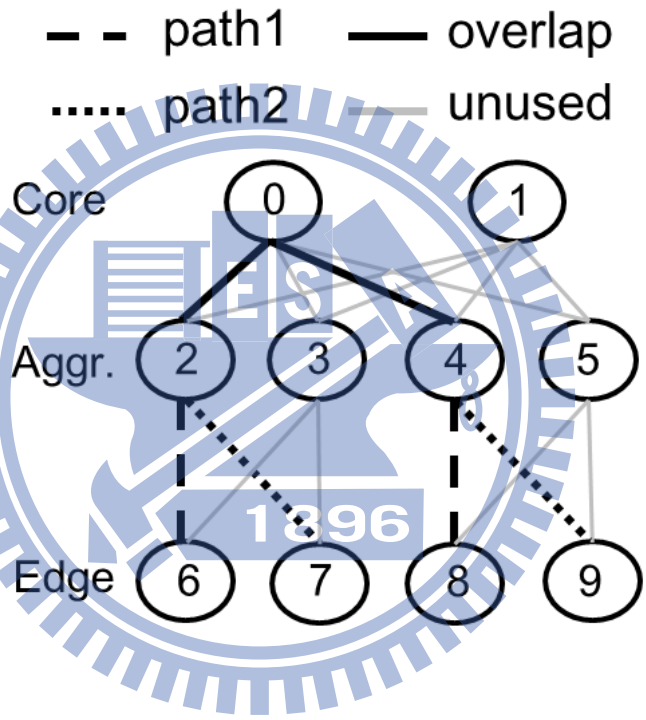


Figure 1.1: STP on fat-tree topology.

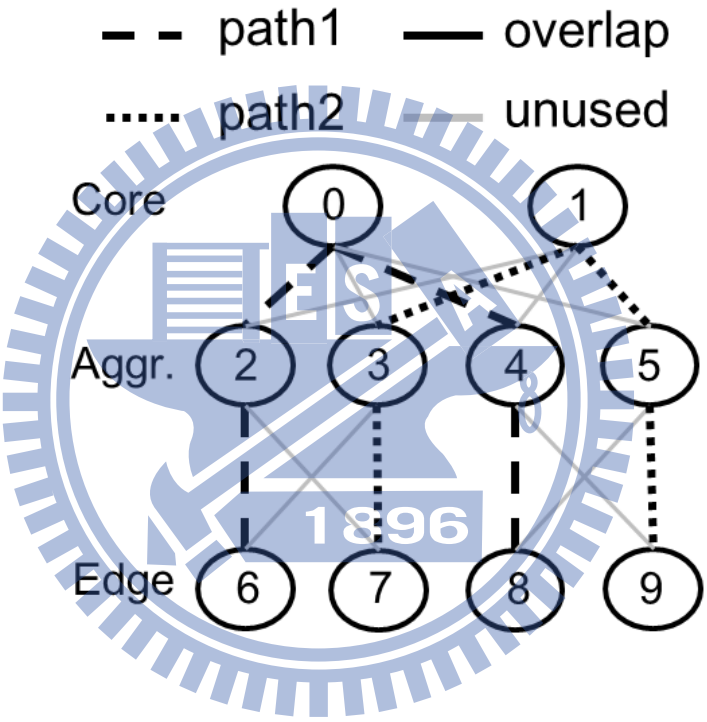


Figure 1.2: D^{2}ENDIST on fat-tree topology.

CHAPTER 2

Background

Network utilization is one of the most critical issues in determining the performance of a datacenter. Various techniques, such as VL2 [1], PortLand [2] and MicroTE [6] are proposed based on multiple-path environment and the control of oversubscription ratio to boost performance. Therefore, when improving network performance of a cloud datacenter, its topology and routing algorithm needs to be considered simultaneously. As a result, two perspectives are elaborated in the following sections.

2.1 Topology

According to [7] and [8], the building cost with high-end switch in DC is high. Owing to the single rooted tree, there are few path selections in traditional DC since it is composed of super computers and high level switches with single root tree . Thus, serious congestion and the link failure may cause the networks to crash because it doesn't exist backup routes. In cloud computing, owing to the parallel computing technique, the topology change more frequently due to link failure and virtual machine (VM) migration. Hence, many new structures of interconnection are proposed for cloud datacenters to improve oversubscription

ratio, multiple-path selection and cost of DC building.

A multi-rooted topology called ***fat-tree*** often serves as the basis of many variants such as VL2 [1], PortLand [2] and Killer Fabric [5]. A fully-mesh structure is proposed in [5] and consists of two stages to support multi-path routing. Because the cost of building a DC is high in the fully mesh networks, an alternative of low-cost networks structures are proposed in [1] and [2]. Due to the lack of scalability on two stage fully mesh networks in [5], three-stages (or more) topology is adopted in [1] and [2]. However, customized hardwares are often required to support the above topologies. Taking [2] as an example, a central control mechanism named **Fabric Manager** is incorporated to map analog medium-access-control (MAC) addresses to Pseudo MAC (PMAC) addresses. In addition, different topology must collocate with specific routing strategy which constrain the design of DC network such as PMAC routing in [2] and BCube Source Routing (BSR) in [3].

2.2 Routing

The layer-3 routing has been widely used in wide area network (WAN) and contains the geographic information in the Internet Protocol (IP). Many layer-3 routing algorithms are proposed such as RIP [9], OSPF [10] and ECMP [11]. Among which, RIP [9] and OSPF [10] are single-path routing which have known to incur poor system throughputs. Besides, in OSPF [10] routing, each individual node (either switch or host) may suffer from the problem of *hello-packet blowup* when the number of the nodes increases, making the computation of each node tremendous.

On the other hand, ECMP [11] is a multi-path routing and intends to effectively utilize the bandwidth of all links. Taking turns to use each link for transmission in ECMP can indeed result in balanced load and better network performance. However, the out-of-order problems of receiving packets cannot be avoided and may incur higher cost on the system.

Moreover, many layer-3 routing mechanisms are adopted because *transmission time* in wide area networks (WANs) is much higher than its encapsulation time. However, this situation is the opposite in local area networks (LANs), particularly in datacenters (DCs) where encapsulation time is the dominant factor. Therefore, layer-2 routing algorithms are often used in DCs [12] [13] , STP [14] [15], SPB [16] [17], ENDIST [4] and HBR [5] [18] are the most prevailing ones. Among all, STP and SPB are the most well-known and easy to implement. STP [14] is a routing algorithm in spanning-tree fashion, whereas SPB [16] is another routing scheme in the shortest-path fashion. However, since both algorithms output fixed single paths for routing, they encounter the same *hello-packet blowup* problem as other layer-3 algorithms.

ENDIST [4] is also a shortest-path like routing algorithm which constructs the network topology first and then computes multiple paths in backbone networks for boosting performance. Although ENDIST is a good candidate for general layer-2 routing, it underlying assumption may not be suitable for DC networks. More specifically, current ENDIST only computes the shortest path between the source and the destination points in WAN. Once one primary path is derived, all other paths are discarded. For example, in Fig. 2.1, for the pair of edge-divided nodes $A1$ and $B1$, $A1-G-B1$ is the shortest path computed by ENDIST. If a node failure happens on node G, the network is down until the node is repaired or the network is reset.

Moreover, when original ENDIST has a tie-break event, the derived paths for different pairs of nodes can be overlapped. For example, in Fig. 2.2, ENDIST may generate paths, $A1-2-0-4-C1$ and $B2-3-0-4-C1$, for communication between (A,C) and (B,C), respectively, where subpath $0-4-C1$ are overlapped, making node 0 and node 4 bottlenecks in routing. As a result, ENDIST remains space for improvement on load balance and thus in our later modification, both static and dynamic weights are added and updated to reflect traffic for boosting performance.

Last, HBR [5] is a method which constructs a routing path as MAC address using

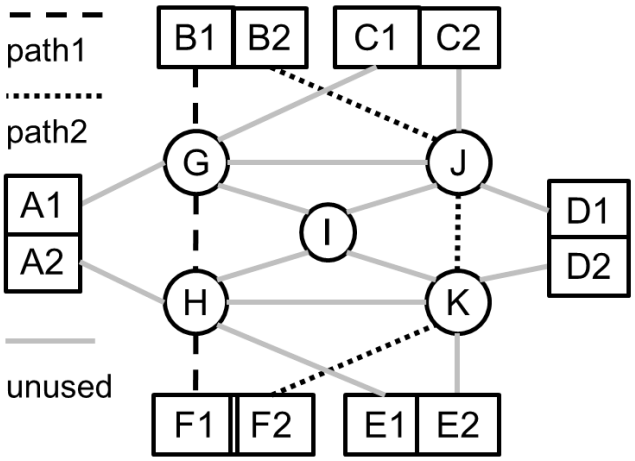


Figure 2.1: ENDIST in WAN.

a hash table. Each packet uses a different uport and decides one of the paths for routing. Many different strategies such as Round Robin, symmetric or asymmetric can be incorporated with HBR. HBR [5] is applied onto a two-stage and fully-mesh topology and thus the packet can be transmitted effectively. However, for multi-stage network topologies, it is not straightforward to construct such a hash table and to apply it for routing.

2.3 Literature Survey

As Fig. 2.3 shown, we can compare the difference between numerical related works. Method [1] proposed a multiple paths calculation algorithm in backbone networks. Method [2], [3], [4] designed different networks topologies in cloud datacenter to solve the oversubscription issue. Method [5] proposed a multiple choices routing scheme in 2-stages cloud datacenter.

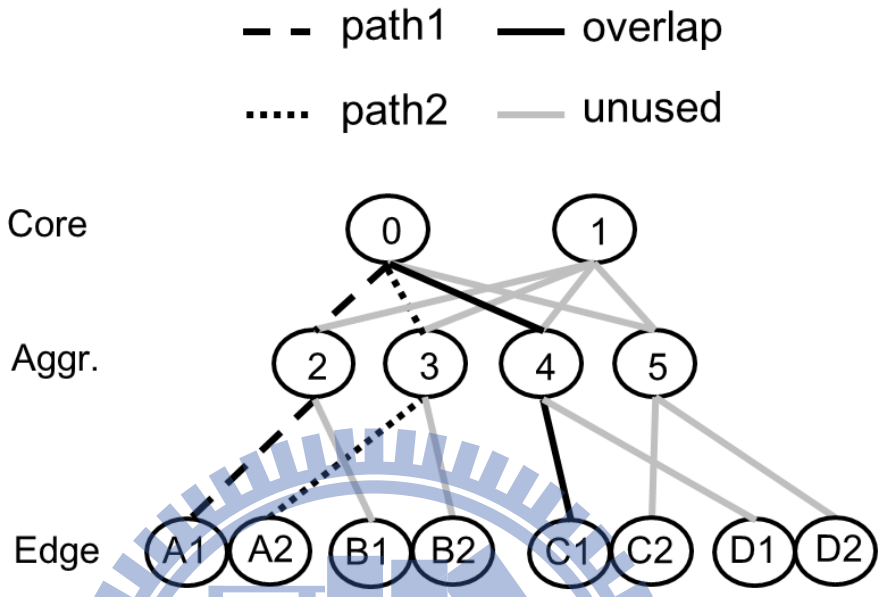


Figure 2.2: ENDIST in DC.

	L2 Routing WAN	L2 Routing DC	L3 Routing	Scalability	Multiple Choice	Load Balancing	Topology Independent
[1] ENDIST, Suh'08	√				√	√	
[2] PortLand, Mysore'09		√		√	√		
[3] VL2, Greenberg'09			√	√	√	√(ECMP)	
[4] BCube, Benson'09			√	√	√	√(Source Routing)	
[5] HBR, Schlansker'10		√			√	√	
Proposal, $D^2ENDIST$'12		√		√	√	√	√

Figure 2.3: Literature survey and paper comparison.

CHAPTER 3

System Model and Problem Formulation

In recent years, many researches focus on the system throughput in cloud datacenter networks. Some of them improve system throughput by changing the networks structure, but their routing protocols can not fully utilize their bandwidth of their structures. Besides, some articles use the existing routing protocol such as Equal Cost Multi-Path Routing (ECMP) which will lead to serious out-of-order problem and makes the system diverge. Hence, we start from the routing research and propose a framework to solve these problems which was already described in Chapter 2.

3.1 System Model

In our scenario, we make use of commodity switches and multi-stages topology. Using commodity switches can reduce the building cost on DC networks. Multi-stage topology can support the multiple path candidates. From top to down in our three-stage topology, we call them cores layer, aggregate layer and edge layer. Among the switches, cores switches and aggregate switches are fully mesh and aggregate switches and edge switches are partial mesh. Fig. 3.1 illustrates the simple three stages topology.

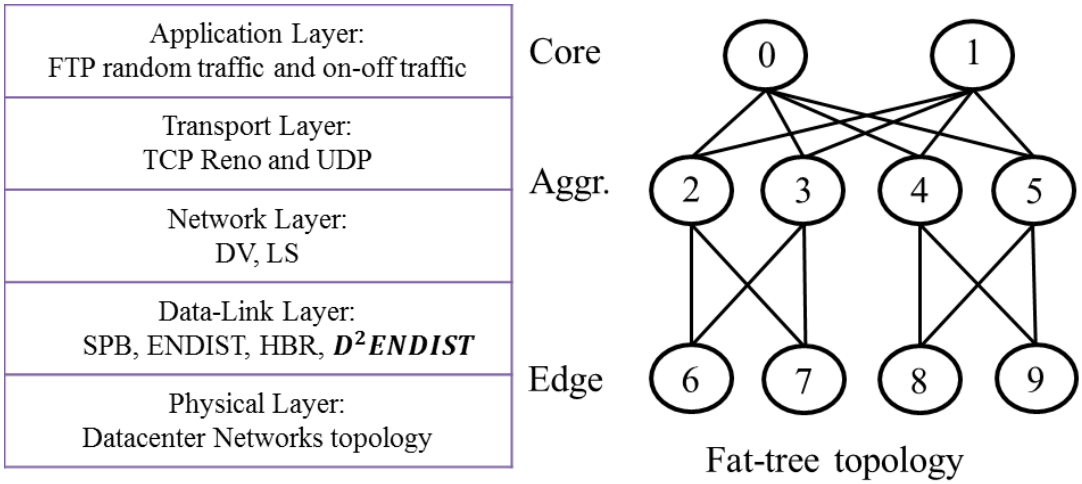


Figure 3.1: System model in NS2 simulation.

Taking Fig. 3.1 as an example, we also consider other parameters in our simulations. In the application layer, we consider TCP random traffic [19] and UDP on-off traffic [20]. In the transport layer, we compare with different mechanisms in [20] and choose TCP Reno as our congestion control strategy. Besides, we also observe the influences of UDP on-off traffic. In the network layer and data link layer, we implement our D2ENDIST routing algorithm and compare with other routing protocols, such as edge node divided spanning tree (ENDIST) [4], hash-based routing (HBR) [5] and ECMP [11]. At last, according to [1], [2], [21] and [22], we set the connection parameters in Physical layer which can minimize the impacts of oversubscription ratio.

3.2 Problem Formulation

In datacenter networks, how to fully utilize the networking resource to avoid system bottleneck is a crucial issue. Since the requests to datacenter grow daily, the amount of computation becomes huge. Besides, owing to the concept of parallel computing, the internal

traffic also plays an important role in DC networks. For this reason, it will cause the system bottleneck easily as we don't dynamic select the adaptive path. Also, if the networks can not work normally, the packet transmitting time of parallel computing will increase which decreases networking efficiency.

Traditional single path routing protocols such as DV and LS routing on particular path which may cause traffic congestion on specific links as Fig. 1.1. Thus, the uneven traffic distribution will lead to low system efficiency as Fig. 1.1. The multipath routing protocol like ECMP will distribute packets to the equal hops paths in round robin way. However, the Round Robin distribution deals with the failure links [23] or congested links badly because all the other links need to wait for the packet. Besides, owing to waiting packets, the receiver must use a buffer to wait the delay packets. If the link still performs a bad transmission, the buffer may grow and lead to system diverse as shown in Fig. 3.2. This is called out-of-order problem. Without packet sequence consideration, it's system performance is quite high. Even so it has good potential, but the out-of-order problem is too serious to implement.

Owing to the VM migration and parallel computing in the intra-networks, the requests from the users of cloud computing in the Internet will cause tremendous traffic. The problem of calculating the minimum cost or minimum overlapping paths in cloud datacenter is an important issue. In this work, we consider the heavy loading in three-stage fat-tree topology networks and propose a routing scheme to solve the aforementioned problem.

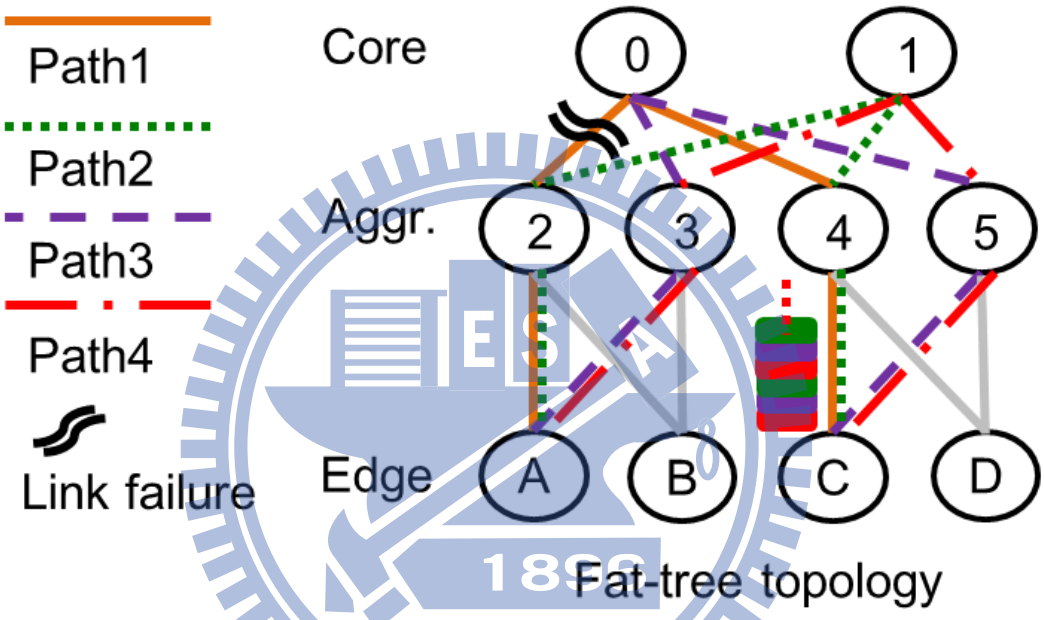


Figure 3.2: Problem description with ECMP. Path1 - Path4 show the equal hops paths on datacenter topology. As one of the equal hops paths becomes congested or failure, the receiver buffer will increase.

CHAPTER 4

Proposed Algorithm: Dynamic and Disjoint Layer-2 Routing

According to the analysis in the previous section, a new routing algorithm is required to support *various types of multi-layer scale network topologies*, *dynamic adjustment of traffic imbalance* and *fast recovery from link failure and VM migration*. In this work, ***dynamic & disjoint ENDIST-based (D^2ENDIST)*** routing algorithm is proposed in Fig. 4.1, consisting of two main stages: (1) routing by *disjoint ENDIST* and (2) rerouting with *dynamic reweights*.

To avoid *overloading* or *underused* problem of interconnection bandwidth in DC networks, Stage 1 of ***D^2ENDIST*** adopts the idea of disjoint paths during a DC network setup to evenly distribute traffic as an initial routing. ENDIST routing [4] serves as the baseline algorithm. However, after a path is formed, the weights in ENDIST keep updating to enable disjoint paths. We design the link with heavy weights to be visited less frequently when computing the next paths. *Backup paths* are computed based on the same idea, accordingly and can fast repair the disconnected DC network caused by link failure.

Stage 2 of ***D^2ENDIST*** is *rerouting with dynamic reweights* where traffic is logged

periodically and provides information to ENDIST for computing paths dynamically. A threshold value is specified by the user, which is for comparing the utilization of each link to determine if rerouting should take place. If the new path is computed, the forwarding table in the central control is updated for the DC network.

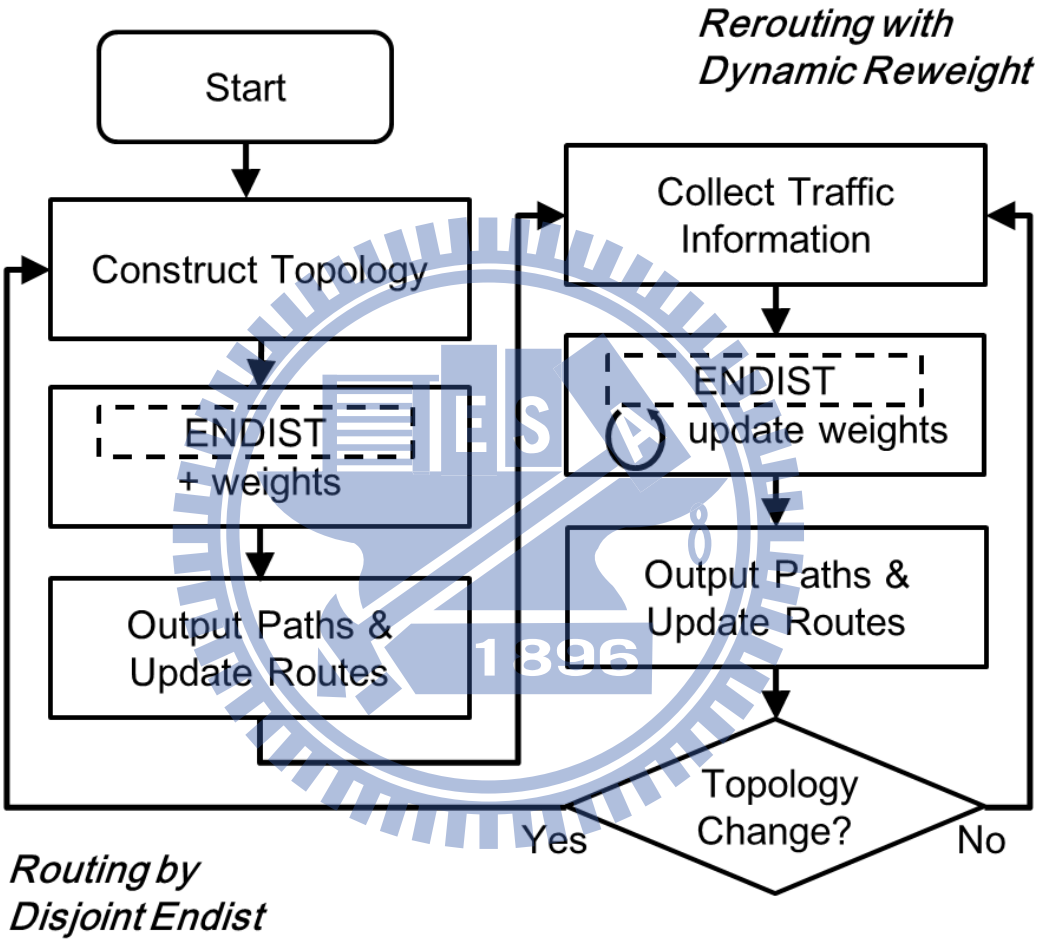


Figure 4.1: Flowchart of ***D^{2}ENDIST routing algorithm***.

4.1 Central Monitor/Control for Cloud Network

A cloud datacenter is typically equipped with a centralized mechanism (central monitor/control) to provide its network performance and tune the networking related components. As the DC encounters a failure at any nodes (either a switch or a host machine) or the network topology changes, recomputing routing paths on the affected nodes occurs. Through the use of Network Management System (NMS) or a similar utility, the central monitor/control can construct the network structure and collect traffic information as shown in Fig. 4.2. The central monitor/control queries all switches and host machines in the DC, and checks the availability of the nodes. A NMS agent in each node replies the connectivity status or the utilization of a link back to the central monitor/control. Once a link is broken or a node crashes, the central monitor/control will be notified and the network topology will be reconstructed.

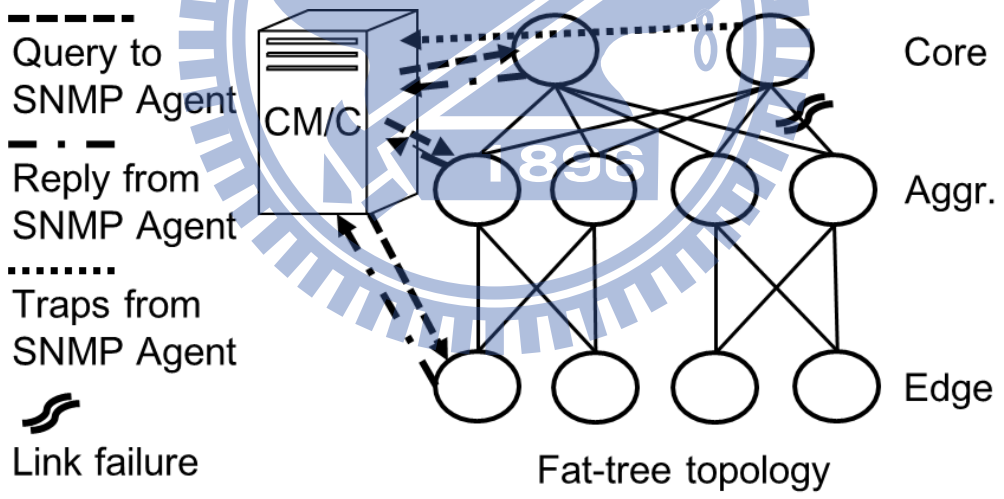


Figure 4.2: Communication between central monitor/control and nodes.

4.2 Disjoint ENDIST Routing

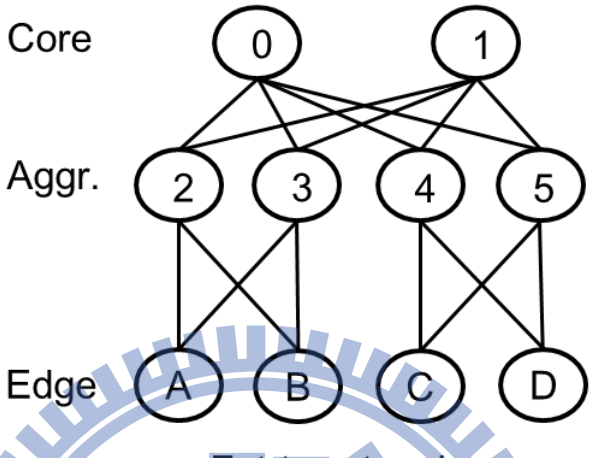


Figure 4.3: Original topology.

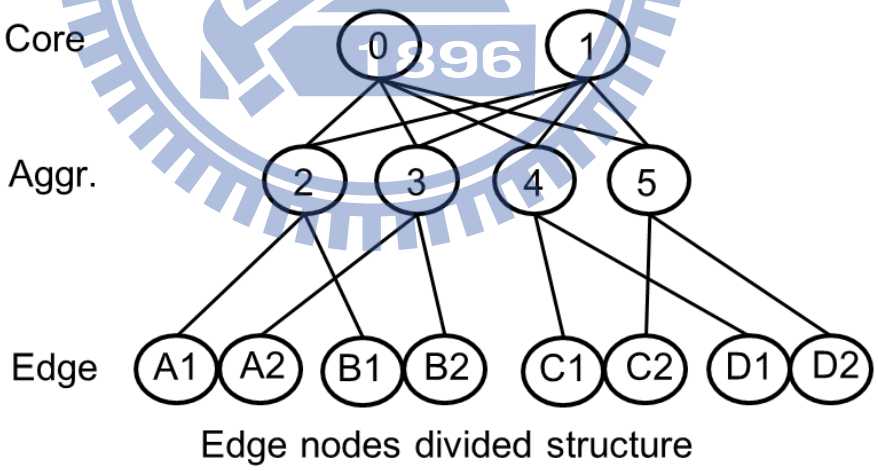


Figure 4.4: ENDIST topology.

Before a DC starting to serve, the route setup can be built on the basis of information

retrieved from the central monitor/control. To maximize link utilization, ENDIST routing [4], as a baseline, is modified to avoid repeated the same path. Since edge-level nodes in the topology are the starting and the ending points in a routing path, our modified ENDIST also replicates a proper number of logical nodes according to the link number on such physical node. In Fig. 4.4, each physical node such as A in the edge level connects two aggregate-level nodes and thus two logical nodes (e.g. $A1$ and $A2$) are generated, resulting in a new topology as shown in Fig. 4.4.

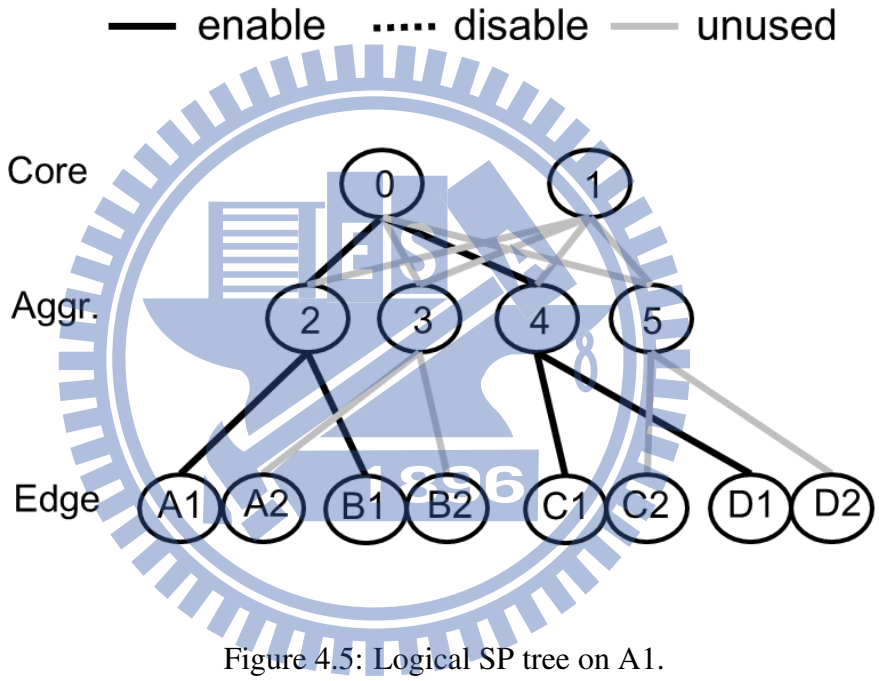


Figure 4.5: Logical SP tree on A1.

Later, our modified ENDIST starts path routing for all physical nodes in order where all weights on links are initialized as 0. To enable disjoint paths, once a path is computed, weights for the links in such path is augmented for computing the next path. Take Fig. 4.3 as an example. A shortest-path algorithm, such as *Dijkstra algorithm* starting from node $A1$ is performed to derive shortest paths (primary paths) to other nodes where each weight of solid lines on these paths is updated as 1 while others remains 0. Disjoint backup paths for recovering link/node failure can be computed by executing Dijkstra algorithm again on

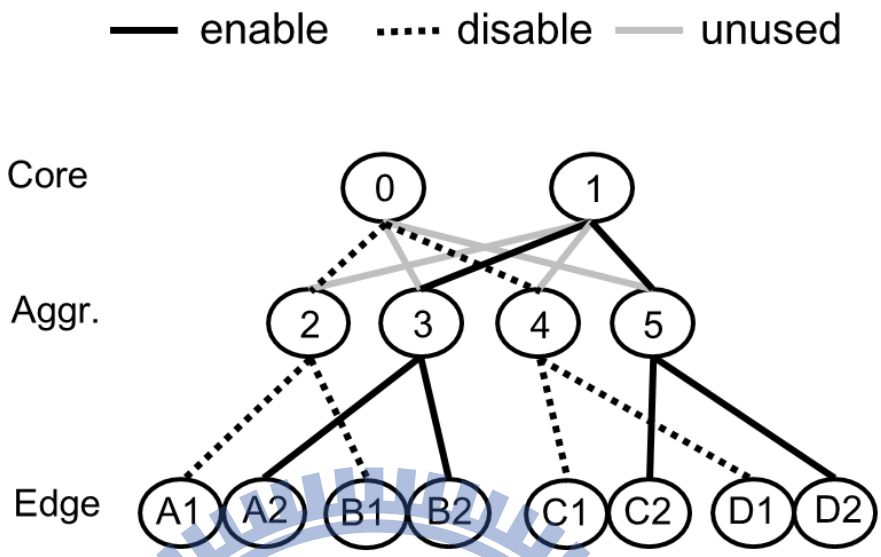


Figure 4.6: Logical SP tree on A2.

node $A2$ and all used links in logical shortest paths for $A1$ are avoided due to the weight update.

Unlike the original ENDIST routing in [4], the disjoint ENDIST (***DENDIST***) routing constructs the shortest-path trees on the logical nodes and needs no extra network cards for a cloud DC. Once disjoint logical shortest-path trees are constructed, nodes and links can be mapped to the physical topology. Figs. 4.7 and 4.8 show the mapping of the disjoint paths generated for nodes $A1$ and $A2$ in Fig. 4.4 to the original topology. As a result, links in physical shortest-path trees for $A1$ and $A2$ are mutually exclusive in this example.

Table 4.1 lists all the primary and backup paths for node 6 in the examples of Figs. 4.7 and 4.8 and will be updated in the central monitor/control of the cloud DC. As one can see, all the primary paths denoted by $6 \overset{p}{\rightsquigarrow} 7$, $6 \overset{p}{\rightsquigarrow} 8$ and $6 \overset{p}{\rightsquigarrow} 9$ are not overlapped with all corresponding backup paths denoted by $6 \overset{b}{\rightsquigarrow} 7$, $6 \overset{b}{\rightsquigarrow} 8$ and $6 \overset{b}{\rightsquigarrow} 9$. Given a source and destination point, a routing path is determined. However, selection strategies can be further

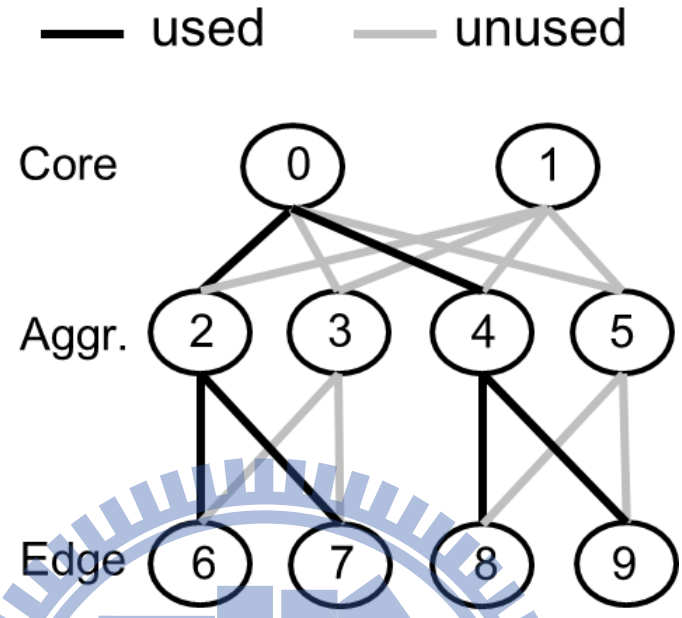


Figure 4.7: Primary SP tree.

divided into asymmetric, symmetric, uniform and single node. Considering different traffic models, performance using different selection strategies may vary. More details can be referred to [5].

4.3 Reroute by Dynamic Reweights

When the DC is operating and traffic transmits continuously, the traffic pattern is time-variant and under-determined. Although *DENDIST* attempts to uniformly distribute traffic through disjoint paths, links of low utilization still cannot be avoided. Considering the instantaneous traffic characteristics, the technique of *dynamic reweighting* is incorporated in *DENDIST*, evolving into ***D*^{2}*ENDIST***.

D*^{2}*ENDIST collects current usage of links as utilization from the central monitor/control and determines if the *DENDIST* routing needs to be performed again for a new

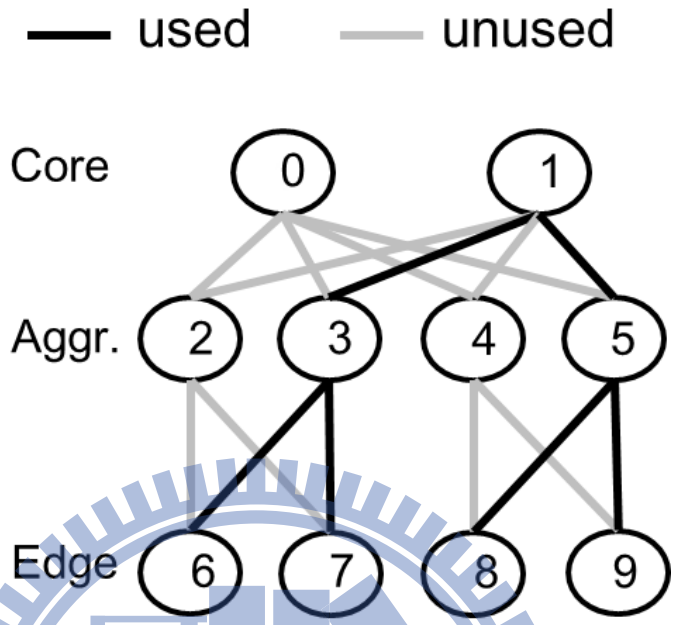


Figure 4.8: Backup SP tree.

routing table. Fig. 4.9 shows the snapshot of the DC network with the traffic loads at time 100 second. Link $2-6$, $2-7$, $4-8$, and $4-9$ in the routing tree for node 6 reach high utilization, which makes nodes 2 and 4 the bottlenecks in the DC network and thus triggers the computation to find a new route. As a result, DENDIST explores the current network and returns another SP tree as shown in Fig. 4.10. Now link $3-6$, $3-7$, $5-8$, and $5-9$ are used to replace old high-utilization links and balance the overall traffic loads.

Different utilization rates represents different weights on links and help trigger re-computation of the routing table if a overloading link is going to appear. However, for a FAT-tree like DC network, links in the upper layer (between core- and aggregate-level nodes) typically have higher bandwidth than those in the lower layer (between aggregate- and edge-level nodes) because of the support of oversubscription. As a result, upper-layer links rarely become bottlenecks in the network. Therefore, for such network, ***D^{2}ENDIST***

Table 4.1: Table including all routing paths for node 6

category	path name	list of nodes
primary	$6 \overset{p}{\rightsquigarrow} 7$	6 2 7
	$6 \overset{p}{\rightsquigarrow} 8$	6 2 0 4 8
	$6 \overset{p}{\rightsquigarrow} 9$	6 2 0 4 9
backup	$6 \overset{b}{\rightsquigarrow} 7$	6 3 7
	$6 \overset{b}{\rightsquigarrow} 8$	6 3 1 5 8
	$6 \overset{b}{\rightsquigarrow} 9$	6 3 1 5 9

can be customized and triggers the recomputation of new routing tables only when a lower-layer link alerts an overloading utilization.

Based on traffic information retrieved from the central monitor/control, *D*^{2}*ENDIST* performs layer-2 routing. Actually, there are multiple choices to decide the final path for given source and destination points. Our *D*^{2}*ENDIST* evaluates each solution and finds the best one as the final path. After deriving all routing paths, a *hashing* technique motivated from [5] is also applied in our *D*^{2}*ENDIST* and records the information about source nodes, intermediate nodes, destination nodes and corresponding weights. Similar with [5], such hash table uses the original network topology, improves the performance and needs no extra network cards like [4]. Fig. 4.11 shows such an example. Assume that nodes 15 and 64 be the source and destination points, respectively. Five different paths connecting node 15 and node 64 exist. Path #4 with the best cost is hashed when transmitting data from node 15 to node 64. There always exist paths in the database of Table 4.2 unless link failure occurs. This table keeps updating the weight values provided by the central monitor/control. The path weights in this table is periodically changed where a bigger weight value typically

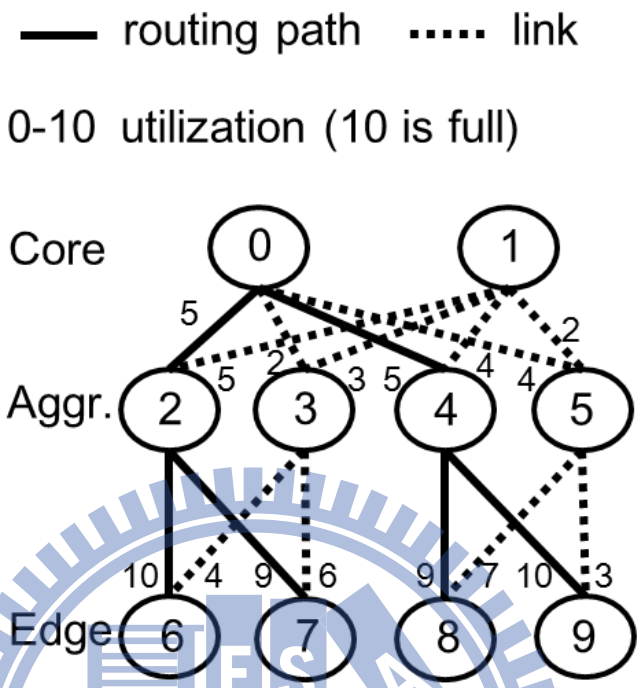


Figure 4.9: Disjoint ENDIST.

represents more traffic. A packet-loss or overloading signal from the DC network will trigger the computation of rerouting and the table renewal in the central monitor/control. Table 4.3 illustrates this situation assuming a packet-loss signal is received. All the routing paths from node 15 to node 64 are recomputed and the hash table is updated, accordingly. As a result, all the paths are renewed and the new path #5 is used to resolve the packet loss signaled from old path #4.

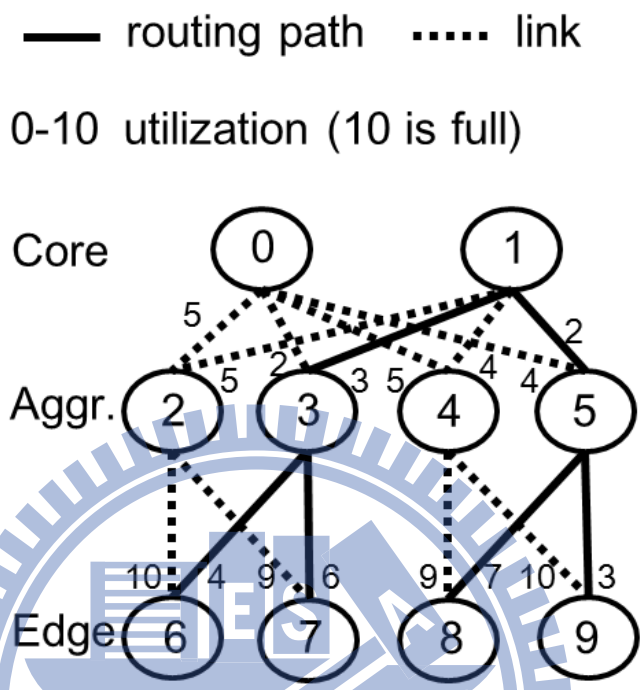


Figure 4.10: Dynamic reroute.

Table 4.2: Path candidates for Hash(15,64) during network initialization

Hash(15,64)	Source	Intermediate		Destination	Weight
Path#1	15	5	0	10	64	9
Path#2	15	6	1	11	64	8
Path#3	15	7	2	12	64	6
Path#4	15	8	3	13	64	5
Path#5	15	9	4	14	64	6

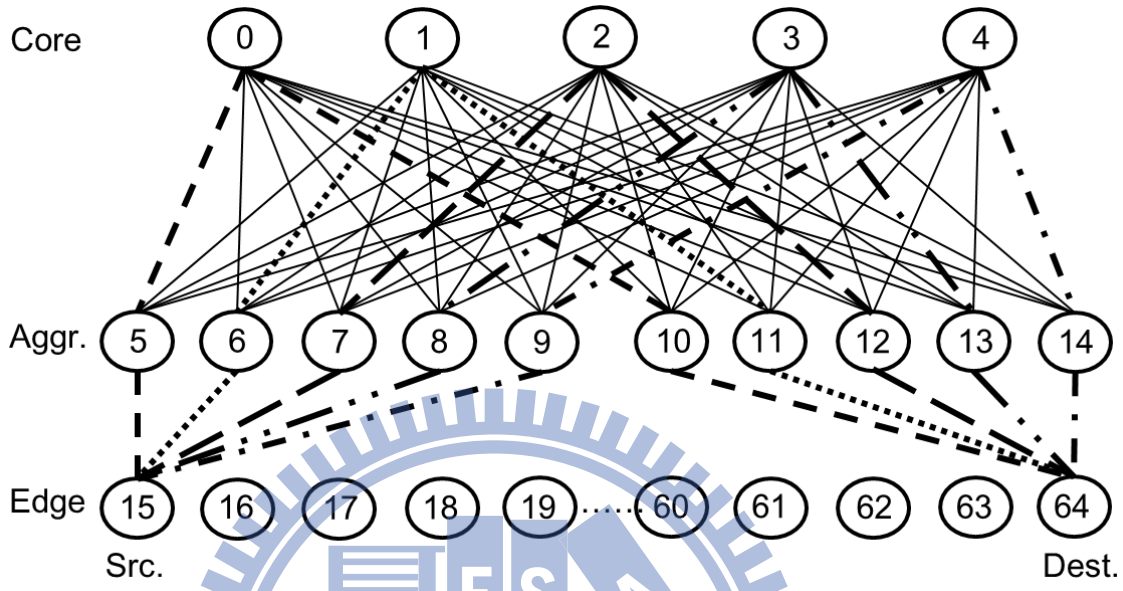


Figure 4.11: Five different paths for routing.

Table 4.3: Updating path candidates for Hash(15,64) at time 100 sec

Hash(15,64)	Source	Intermediate			Destination	Weight
Path#1	15	5	4	13	64	6
Path#2	15	6	2	14	64	6
Path#3	15	7	3	12	64	6
Path#4	15	8	3	14	64	7
Path#5	15	9	1	12	64	3

CHAPTER 5

Numerical Results

A 5-10-50 topology consisting of 5 core-level, 10 aggregate-level and 50 edge-level nodes is mainly extracted from an industrial datacenter network and used in our experiments. The bandwidth between core-aggregate and aggregate-edge link is 10:1 where the core-aggregate nodes are fully mesh and the aggregate-edge are semi-fully mesh. Traffic loads in our experiments are heavy and simulation runs up to 200 second to obtain the stable performance. The simulation tool is Network Simulator-2 (NS2) [24] and the parameters are set as Fig. 5.1.

5.1 Throughput under Various Traffic Models and Topologies

To correctly evaluate the performance of a DC network, traffic characteristics are of concern according to [20] and [25]. Particularly, the bottleneck of a fat-tree DC network often locates the aggregate-level nodes and the behaviour of traffic can be described by a log-normal on-off model. Therefore, both TCP traffic and UDP traffic modelled by the log-normal on-off model are used to conduct experiments for comparing throughout.

Parameter	Value
Topology	Fat-tree
Rerouting Period	5 seconds
Routing Path Calculation Method	Disjoint ENDIST
Overload threshold	90% ↑
Rerouting threshold	50% ↓
Flow Numbers	1000
Flow Size	1M bytes

Figure 5.1: The traffic and environment parameters in NS2.

The first target topology extracted from an industrial design is a three-layer fat-tree semi-mesh network and composed of 5 core-level, 10 aggregate-level and 50 edge-level switches. We perform layer-two SPB, ENDIST, HBR and our D^{2}ENDIST routings and compare their throughputs. As a result, *D^{2}ENDIST* outperforms the others and achieves 10% more throughput than the second best HBR under the TCP traffic model in Fig. 5.2. As to on-off log-normal UDP traffic, the final throughput of *D^{2}ENDIST* is 43 Kbps, showing 6.9% percent improvement to HBR in Fig. 5.3.

Besides, we implement on different topology such as PortLand and multi-level (more than three levels) structure. In Fig. 5.4, one can see that D^{2}ENDIST achieves 26.5% improvement than ENDIST and 81.9% improvement than the default routing of PortLand. In addition, we can observe the the same trend in the multi-levels topology in Fig. 5.5.

The effects of topology change like VM migration and link failure will send the trap information actively and trigger the network monitor server, which drives the server to re-calculate the new path of the influence nodes. Besides, the convergence time is quite

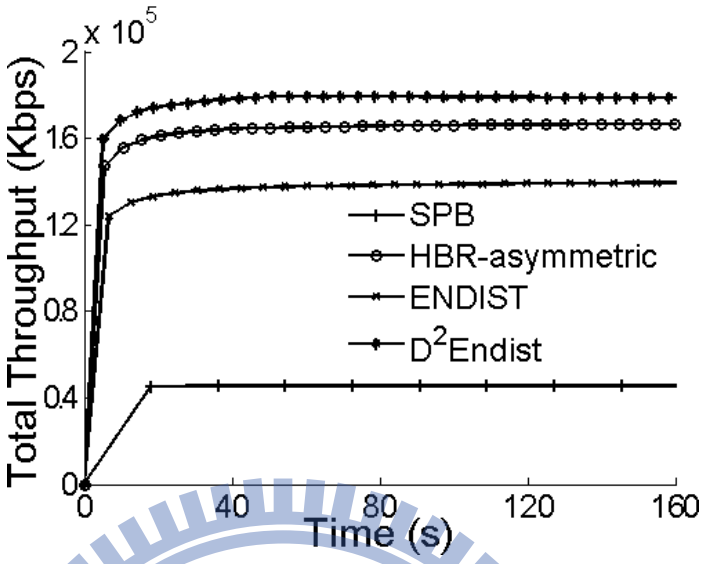


Figure 5.2: Throughput comparison of different routing algorithms in a fat-tree topology with TCP traffic model.

small in our experiment.

5.2 Failure Recovery Time

Since the central monitor/control records the traffic information and topology change, *D*^{2}*ENDIST* can retrieve traffic data accordingly. In our second experiment, we investigate the required time that *D*^{2}*ENDIST* can recompute the routing network for a random link failure and resume communication. Fig. 5.6 shows that 0.06 second is needed to fix the link failure and recover the network, where the x-axis denotes the time and y-axis denotes the packet index.

Besides, we can observe the effect of link failure in Fig. 5.7. If we increase the number of failure links, the system throughput will drop as the ratio of equal hops paths does. In D^{2}ENDIST, the remaining nodes exist sufficient path to support disjoint paths, and

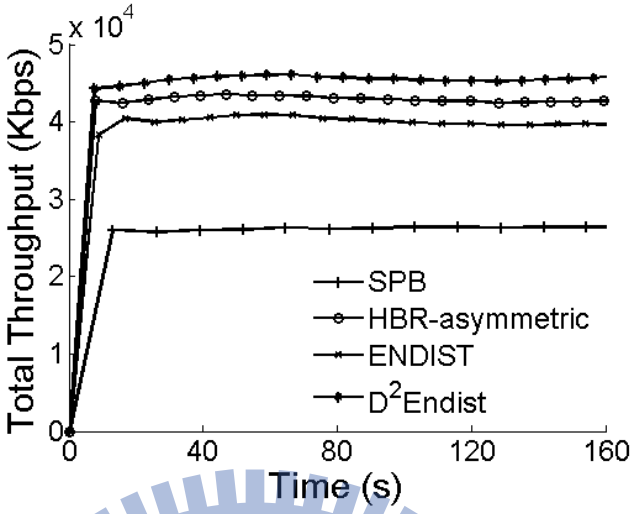


Figure 5.3: Throughput comparison of different routing algorithms in a fat-tree topology with UDP traffic.

thus the influence of link failure is small.

5.3 Delay and Link Utilization

The objective of rerouting is to balance the link utilization between different paths. Based on the idea, the traffic information on our fat-tree topology and the packet delay are updated periodically. In our third experiment, we study the impact of different values of reroute periods on the total throughputs. Figs. 5.8 and 5.9 illustrate this comparison. As a result, within 200 second simulation, the more frequently the traffic information is updated, the higher the average throughput can be obtained. Fig. 5.9 shows that $D^2ENDIST$ can achieve comparable or better performance than the equal-cost multi-path (ECMP) routing. Last but not least, a proper value for the update period in a real DC network still depends on data collection, TCP re-ack and other factors, which must be calibrated carefully.

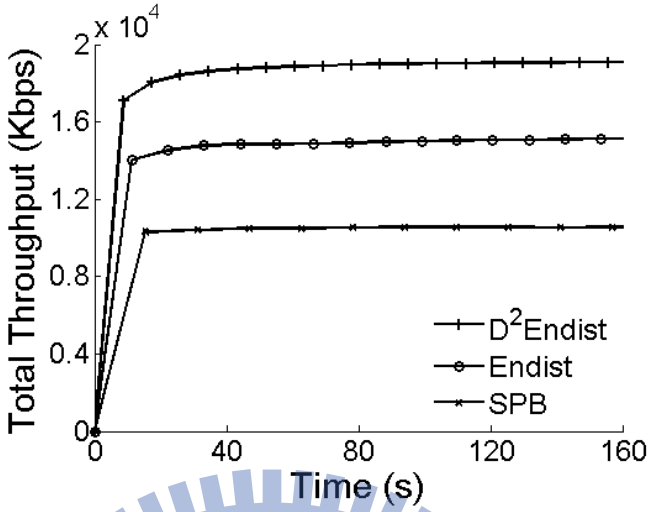


Figure 5.4: Throughput comparison of different routing algorithms in a PortLand topology with TCP traffic model.

Given a reroute period, we would like to further investigate the effectiveness of *D*^{2}*ENDIST* on improving link utilization. Fig. 5.10 shows the distributions change on link utilization between ENDIST and D^{2}ENDIST. As one can see, link utilization for ENDIST is almost linear. The link utilization of whole links are close. After using the disjoint paths and dynamic reroute in D^{2}ENDIST, the cumulative distribution function (CDF) shows almost 60% (40% to 100%) of the links between 80-100% utilization.

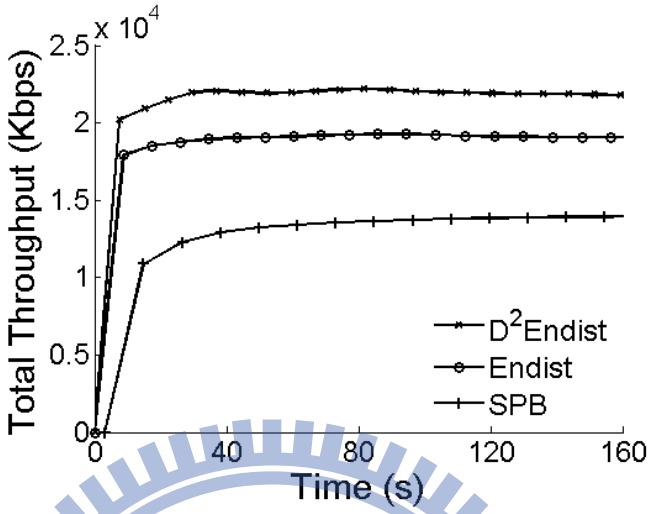


Figure 5.5: Throughput comparison of different routing algorithms in a multi-stage topology with TCP traffic.

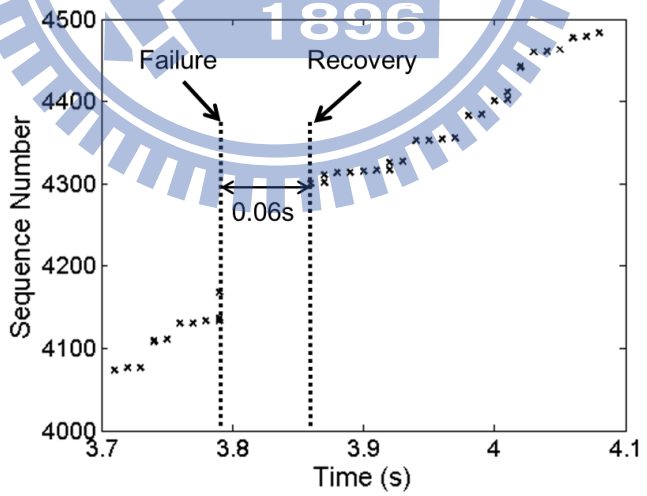


Figure 5.6: Recovery time for the network topology.

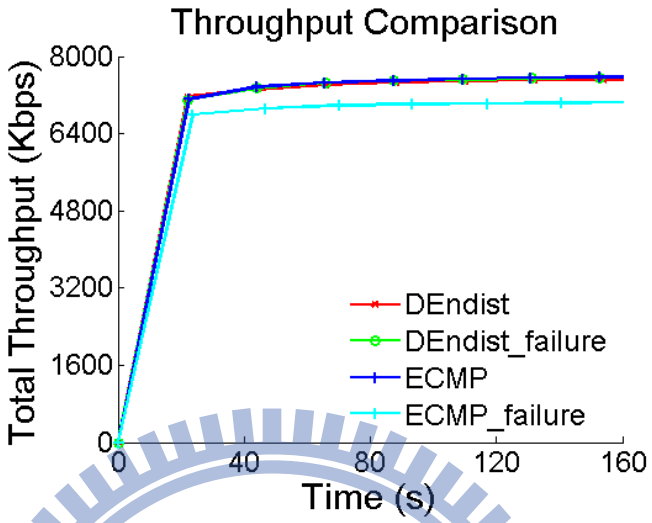


Figure 5.7: The throughput comparison of different routing schemes under link failure.

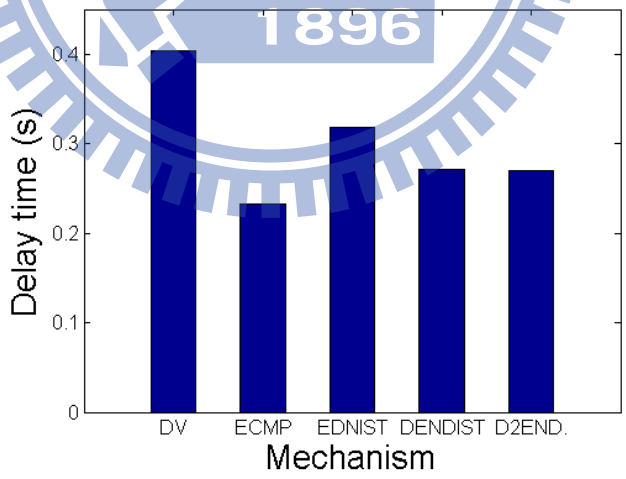


Figure 5.8: Average packet delay of different routing schemes.

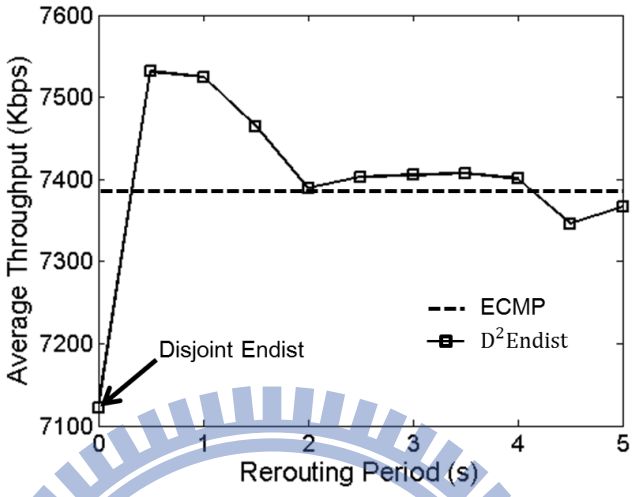


Figure 5.9: Throughput comparison with different re-routing period.

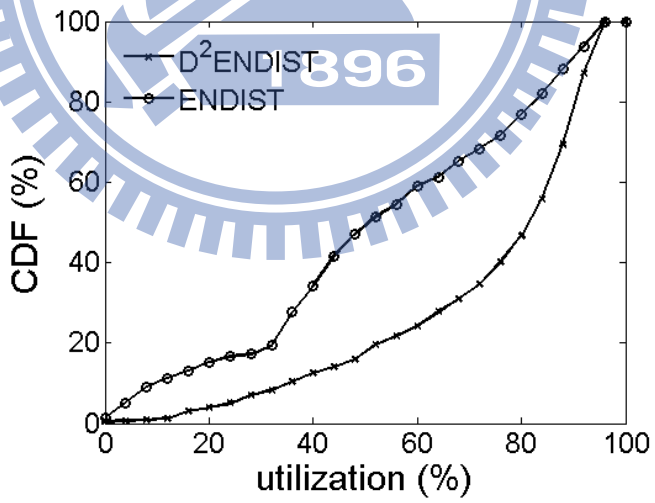


Figure 5.10: CDF Comparison of link utilizations.

CHAPTER 6

Conclusions

6.1 Summary

The routing strategy is a critical issue in datacenter (DC) networks. Owing to the parallel computing and VM migration take place frequently, how to maintain effective DC networks condition is an important issue. Many previous works proposed a framework of DC design to improve the single path environment and reduce the cost of DC deployment. However, they still suffer from the *overloading* or *underused* problems of interconnection bandwidth in DC networks. Therefore, in this thesis, we the propose the D^{2}ENDIST routing algorithm.

Experiments show that our approach can provide better system throughput than other routing strategy. D^{2}ENDIST outperforms other routing schemes and achieves 10% more throughput than the second best HBR under the TCP traffic model. As for on-off log-normal UDP traffic, the final throughput of D^{2}ENDIST shows 6.9% percent improvement compared to HBR. Also, D^{2}ENDIST only takes 0.06 second to fix the link failure and recover the network. The recovery time from link failure and VM migration. Besides, D^{2}ENDIST can achieve better performance than ECMP when the traffic information is

updated more frequently. At last, when implementing other fat-tree topologies, D^{2}ENDIST outperforms other routing strategies. As a result, our D^{2}ENDIST successfully demonstrates its effectiveness on fat-tree like topology.

6.2 Future Work

The following issues are worth-while for further investigation:

1. Service Level Agreement Problem:

   The delay tolerance varies from different applications such as data integrity or real time streaming traffic. Besides, the rent of enterprise may separate the networks into many sub-networks. We will consider the QoS or Service Level Agreement (SLA) to serve users by adapting resource allocation or find a better trade between system efficiency and power consumption. Thus, the customers of cloud computing can choose suitable scheme in flexibility.

2. Out-of-Order Problem:

   We observe the out-of-order problem [26] in equal hops networks topology. Hence, in our routing design, we also consider this problem. We will evaluate the throughput of out-of-order problem based on our simulation platform to illustrate this issue. However, the architecture in NS2 not easy to simulate Automatic Repeat-reQuest (ARQ), which makes the simulation of the out-of-order problem in NS2 platform become challenging.

3. Mathematical Model:

   In our DC networking framework, we are still lack of mathematical analysis. Hence, we want to derive a mathematical model to promote the reliability. Besides, we can prove the property of disjoint algorithm and acquire the optimal factors from math model.

Bibliography

[1] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," in *ACM SIGCOMM on Data communication*. New York, NY, USA: ACM, pp. 51–62, Aug 2009.

[2] R. Niranjan Mysore, A. Pamboris, N. Farrington, N. Huang, P. Miri, S. Radhakrishnan, V. Subramanya, and A. Vahdat, "PortLand: a scalable fault-tolerant layer 2 data center network fabric," in *ACM SIGCOMM on Data communication*. New York, NY, USA: ACM, pp. 39–50, Aug 2009.

[3] 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," in *ACM SIGCOMM on Data communication*. New York, NY, USA: ACM, pp. 63–74, Aug 2009.

[4] C. Suh, K. Kim, and J. Shin, "ENDIST: Edge node divided spanning tree," in *10th International Conference on Advanced Communication Technology*, vol. 1, pp. 802 –807, Feb 2008.

[5] M. Schlansker, J. Tourrilhes, Y. Turner, and J. Santos, "Killer fabrics for scalable datacenters," in *IEEE International Conference on Communications*, pp. 1 –6, May 2010.

[6] T. Benson, A. Anand, A. Akella, and M. Zhang, "MicroTE: fine grained traffic engineering for data centers," in *Conference on emerging Networking Experiments and Technologies*. New York, NY, USA: ACM, pp. 8:1–8:12, Dec 2011.

[7] M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in *ACM SIGCOMM 2008 conference on Data communication*. New York, NY, USA: ACM, pp. 63–74, Aug 2008.

[8] A. Greenberg, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "Towards a next generation data center architecture: scalability and commoditization," in *ACM workshop on Programmable routers for extensible services of tomorrow*. New York, NY, USA: ACM, pp. 57–62, Aug 2008.

[9] C. Hendrick, "Routing information protocol," IETF, RFC 1058, June 1988.

[10] J. Moy, "OSPF version 2," IETF, RFC 1583, March 1994.

[11] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast next-hop selection," RFC 2991, Nov. 2000.

[12] C. Suh, Z. Luo, D.-Y. Kim, and B.-S. Joo, "An analytic model for spanning tree based routing methods," in *11th International Conference on Advanced Communication Technology*, vol. 01, pp. 605 –610, Feb 2009.

[13] C. Suh and S. woong Jung, "Layer-2 routing analytic model by linear programming," in *10th International Conference on Advanced Communication Technology*, vol. 1, pp. 561 –566, Feb 2008.

[14] *IEEE Standard for Local and metropolitan area networks Media Access Control (MAC) Bridges*, IEEE Std. 802.1D, 2004.

[15] H. T. Viet, Y. Deville, O. Bonaventure, and P. Francois, "Traffic engineering for multiple spanning tree protocol in large data centers," in *International Teletraffic Congress*, pp. 23 –30, Sep 2011.

[16] *Virtual Bridged Local Area Networks, Amendment 9: Shortest Path Bridging*, IEEE Std. 802.1aq/D0.3, 2006.

[17] D. Allan, P. Ashwood-Smith, N. Bragg, J. Farkas, D. Fedyk, M. Ouellete, M. Seaman, and P. Unbehagen, "Shortest path bridging: Efficient control of larger ethernet networks," *IEEE Communications Magazine*, vol. 48, pp. 128–135, 2010.

[18] M. Schlansker, Y. Turner, J. Tourrilhes, and A. Karp, "Ensemble routing for datacenter networks," in *ACM/IEEE Symposium on Architectures for Networking and Communications Systems*. New York, NY, USA: ACM, pp. 23:1–23:12, Oct 2010.

[19] L. A. Grieco and S. Mascolo, "Performance evaluation and comparison of westwood+, new reno, and vegas tcp congestion control," *SIGCOMM Comput. Commun. Rev.*, vol. 34, no. 2, pp. 25–38, Apr. 2004.

[20] T. Benson, A. Anand, A. Akella, and M. Zhang, "Understanding data center traffic characteristics," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 92–99, Jan. 2010.

[21] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, pp. 75–86, Aug. 2008.

[22] S. Kandula, J. Padhye, and P. Bahl, "Flyways to de-congest data center networks," in *Association for Computing Machinery, Inc*, Oct. 2009.

[23] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in *ACM SIGCOMM 2011*. New York, NY, USA: ACM, pp. 350–361, Aug 2011.

[24] *Network simulator ns-2 http://www.isi.edu/nsnam/ns*.

[25] S. Kandula, S. Sengupta, A. Greenberg, P. Patel, and R. Chaiken, "The nature of data center traffic: measurements & analysis," in *ACM SIGCOMM on Internet Measurement*. New York, NY, USA: ACM, pp. 202–208, Nov 2009.

[26] J. Kim and B. Ahn, "Next-hop selection algorithm over ecmp," in *Asia-Pacific Conference on APCC 2006*, pp. 1 –5 Aug 2006.

Vita

Gen-Hen Liu was born in Taiwan, R. O. C. in 1988. He received a B.S. in Electrical and Computer Engineering from National Chiao-Tung University in 2010. From July 2010 to September 2012, he worked his Master degree in the Mobile Communications and Cloud Computing Lab in the Department of Communication Engineering at National Chiao-Tung University. His research interests are in the field of Cloud Computing.