British Journal of Mathematics & Computer Science
9(1): 1-11, 2015, Article no.BJMCS.2015.182
ISSN: 2231-0851

SCIENCEDOMAIN *international*
www.sciencedomain.org

Architecting Network Structure for Data Center: A Scalable Hierarchical Approach

Biao Dong[1*]

[1]*Department of Computer and Software, Nanjing Institute of Industry Technology, Nanjing, China.*

Article Information

DOI: 10.9734/BJMCS/2015/17149
Editor(s):
(1) Qiang Duan, Information Sciences & Technology Department, The Pennsylvania State University, USA.
Reviewers:
(1) Anonymous, KCL Institute of Management and Technology, India.
(2) Ahmet Sayar, Computer Engineering Dept., Kocaeli University, Turkey.
Complete Peer review History: http://www.sciencedomain.org/review-history.php?iid=1142&id=6&aid=9216

Original Research Article

Received: 27 February 2015
Accepted: 15 April 2015
Published: 11 May 2015

Abstract

This paper presents a server centric approach for architecting data center network(DCN) by using a hierarchical model. Considering that the network infrastructure must be scalable to a large number of servers and allow for incremental expansion, we design a high scalable DCN with high performance. The results imply that our approach is more feasible, and possess the good regularity and expandability that help reduce the cost of further expansions.

Keywords: Data center network; topology; routing; parallel paths.

1 Introduction

As web search, e-commerce, data storage, video online, high-performance computing, data analysis and other information services develop toward socialization, dynamization and centralization, applications, computing and storage resources on Internet are migrating to the data center. DCN is a core component of the data center, it is responsible for the interconnection of tens or hundreds of thousands of servers, and provides efficient network communication and data transmission capabilities for the upper computing services. That DCN faces to the challenge of the application environmental change makes it different from the traditional interconnection networks, such as Ethernet, grid and high performance distributed computing system, in the fields of design requirement, construction of topology, application environment and evaluation standard. The new application service models represented by cloud computing put forward new requirements for DCN performances in scalability, and fault tolerance.

From the point of view of DCN application environments and its own technological evolutions, we must deal with some new research problems and needs that are identified throughout DCN.

**Corresponding author: dongb@niit.edu.cn, db@niit.edu.cn;*

(1) DCN topology. Through network cable, and according to certain rules of interconnection, servers and switches are connected to form a specific topology. DCN is a fundamental problem of data center. Today, the total amount of data processing and storage of data center have been increasing at a rapid rate. The data storage volume for now is already PB-level, and the number of node servers reach hundreds of thousands or even millions. With the rapid growth and large-scale deployment, DCN's structure has become increasingly complex and DCN's size is expanding. For existing DCN expansion, maintenance, reconstruction and cost, these are a great challenge. Therefore, we need a new kind of DCN topology which has the following characteristics, such as low cost, easy build and expansion, and simlpe maintenance and wiring.
(2) Fault tolerant routing. The node servers of DCN are not only used to process and store data, but also to participate in forwarding and routing. In order to reduce the construction cost, DCN generally adopts the low price commercial servers. This makes in node failure normalization. When a failed node is used as an intermediate node, routing can not be forwarded. Therefore, DCN need a fault tolerant routing mechanism, so as to efficiently complete routing in the case of intermediate node failures.

In response to these new demands, the paper researches on DCN topology, fault-tolerant routing, and intends to solve the underlying technical problems and actual application deployment challenges caused by the rapid development of new applications.

The rest of the paper is organized as follows. In Section 2, we review some related works. Our main methods including sparse hierarchical graph (SHG) networks structure, and routing algorithm in SHG are presented in Section 3 and Section 4, respectively. The performance evaluations are given in Section 5. Finally, the paper is concluded in Section 6.

2 Related Work

In order to solve this problem and put our work in perspective, we give a brief overview of related works. A major approach is switch centric, which organizes switches into structures other than tree and puts the interconnection intelligence on switches. Al-Fares M et al. [1] leverage largely commodity Ethernet switches to support the full aggregate bandwidth of clusters consisting of tens of thousands of elements. Based on the fat-tree, they present techniques to perform scalable routing while remaining compatible with Ethernet, IP, and TCP. Mysore R N et al. [2] observe that in DCN, the baseline multi-rooted topology is known and relatively fixed, and leverage this observation in the design of Port L and, a set of Ethernet-compatible routing, forwarding protocols specically tailored for data center deployments. Heller B et al. [3] examine the trade-offs between energy efficiency, performance and robustness, and present Elastic Tree, a network-wide power manager, which dynamically adjusts the set of active network elements (links and switches) to satisfy changing data center traffic loads. VL2 uses flat addressing, valiant load balancing, and end system-based address resolution, to support huge data centers with uniform high capacity between servers, performance isolation between services, and Ethernet layer-2 semantics [4]. Helios is a hybrid electrical/optical switch architecture that can deliver significant reductions in the number of switching elements, cabling, cost, and power consumption relative to recently proposed data center network architectures [5]. Wang G et al. [6] propose a hybrid packet and circuit switched DCN architecture, namely c-Through, which augments the traditional hierarchy of packet switches with a high speed, low complexity, rack-to-rack optical circuit-switched network to supply high bandwidth to applications. OSA [7] is an optical switching architecture for DCN. Leveraging runtime reconfigurable optical devices, OSA dynamically changes its topology and link capacities, thereby achieving unprecedented flexibility to adapt to dynamic traffic patterns [7]. The above Helios, c-Through and OSA belong to hybrid optical-electric switching scheme. [7] and WDCN [8,9] are presented that the data center environment is well suited to a deployment of 60 GHz links contrary to concerns about interference and link reliability, and explore its use to relieve hotspots in oversubscribed DCN.

Another relevant method is server centric, which puts the interconnection intelligence on servers and uses switches only as cross bars. Guo D et al. [10] present HCN, the structure for data centers owning the advantages of expansibility and equal degree. HCN offers high degree of regularity, scalability and symmetry which very well conform to the modular design and implementation of data centers [10]. MDCube [11] is a high performance interconnection structure to scale its containers to mega-data centers, and uses the high-speed up-link interfaces of the commodity switches in the containers to build the inter-container structure, reducing the cabling complexity. To alleviate the growing concern of energy waste in networked devices, Huang L et al. [12] present PCube, a server-centric data center structure that conserves energy by varying bandwidth availability based on traffic demand. Kliazovich D et al. [13] underline the role of communication fabric in data center energy consumption and present a methodology, termed DENS, that combines energy-efficient scheduling with network awareness [13]. Leveraging the introduction of all-optical switching technologies inside the data center, LIGHTNESS [14] aims at realizing a flexible and scalable DCN solution featuring ultra-high data throughput and low-latency server-to-server communication. Ghosh A et al. [15] investigate two semi-centralized designs that lie at practical points along the spectrum between fully-distributed and fully-centralized solutions, and achieve scalability by distributing computation across multiple tiers of optimization machinery. By exploring smart grid technologies that can be applied to a telecommunication network to achieve energy-efficient data center networking, Koutitas G et al. [16] establish an active role in the energy market by adjusting power consumption in real time. Singla A et al. [17] present the first non-trivial upper bound on network throughput under uniform traffic patterns for any topology with identical switches, and show that random graphs achieve throughput surprisingly close to this bound.

There are three challenges for DCN. First, the network infrastructure must be scalable to a large number of servers and allow for incremental expansion. Second, DCN must be fault tolerant against various types of server failures, link outages, or server-rack failures. Third, DCN must be able to provide high network capacity to better support bandwidth-hungry services. Existing switch centric structures cannot support one-to-x traffic well and need upgrading switches. Existing server centric structures either cannot provide high network capacity or use a large number of server ports and wires.

Inspired by our previous work [18], this paper proposes a scalable hierarchical approach for architecting network structure for data center, and addresses two issues:

(1) The establishment of SHG that allow for DCN incremental expansion.
(2) The design and implementation of routing algorithm in SHG that include shortest path routing and fault-tolerant routing.

Compared with other works on DCN, the major contributions of this paper are summarized as follows.

(1) Switch centric approaches need upgrading switches to support for DCN incremental expansion. Because SHG belongs to server centric approaches, the deployment of new SHG topology is more feasible, and SHG topology can be used to improve the end-to-end throughput.
(2) Server centric approaches usually use more than two ports per server to scale to a large server population. Because SHG is sparse graph with the server degree 2, SHG possess the good regularity and expandability that help reduce the cost of further expansions.

3 The SHG Networks Structure

3.1 Basic Concept and Definitions

SHG is an approach for modeling and implementing the DCN. The concepts associated with SHG are defined as follows.

Definition 3.1. Let $\chi=\{\sigma_i|-1\le i\le n\}$ be an alphabet, a non-empty finite set. A regular identifier (RI) over χ is a finite sequence of elements from χ, and satisfies one of two ways.

(1) σ_{-1} is a regular identifier.
(2) $\sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_p}$ is a regular identifier, for any $\{\sigma_{i_1},\sigma_{i_2},\cdots,\sigma_{i_p}\}\subseteq\chi-\{\sigma_{-1}\}$, where $p(p\ge 1)$ is integer number, and $i_1<i_2<...<i_p$.

Some notations concerning the RI are as follows. First, $RI(p)=\sigma_1\sigma_2\ldots\sigma_p$, $RI(<p)=\sigma_1\sigma_2\ldots\sigma_{p-1}$, $RI(>p)=\sigma_{p+1}\sigma_{p+2}\ldots\sigma_n$, where $p(p\ge 1)$ is an integer number, Similarly, $RI(<i,\ge j)=\sigma_{i-1}\circ\sigma_{i-2}\circ\ldots\circ\sigma_j$. Second, let $\chi^*=\chi^0\cup\chi^1\cup\cdots$, where $\chi^0=\{\sigma_{-1}\},\chi^1=\chi-\{\sigma_{-1}\}$, $\chi^i=\{RI(i)|\ i>1\}$. Third, given an arbitrary RI, its alphabet is denoted by $\chi(RI)$. Its length is the number of elements in RI, and is denoted |RI|. Finally, given two arbitrary RI_1 and RI_2, they are called the adjacent RIs iif $|RI_1|-|RI_2|=1$, or $|RI_1|-|RI_2|=-1$.

Definition 3.2. Given two arbitrary RIs RI_1 and RI_2, multiplication is signified by the circle sign($\circ$), $RI_1\circ RI_2=\sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_p}$, where $\{\sigma_{i_1},\sigma_{i_2},\cdots,\sigma_{i_p}\}=\chi\{RI_1\}\bigcup\chi\{RI_2\}$, and $\sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_p}$ is a RI. Subtraction is signified by the minus sign(-).RI_1-RI_2 represents the operation of removing all characters which are not in $\chi-\chi(RI_2)$.

Definition 3.3. A complete hierarchical graph (CHG) over χ, denoted by $CH_{|\chi|}(V,E)$ (also called $CH_{|\chi|}$ for short), is an undirected graph with $V=\chi^*$, and then $(RI_1, RI_2)\in E$ if and only if RI_1 and RI_2 are adjacent.

Definition 3.4. A SHG over χ, denoted by $SH_{(M,|\chi|)}(V,E)$, also called $SH(M,|\chi|)$ for short, comprises $M\in(2^{|\chi|-1}, 2^{|\chi|})$ nodes, is recursively defined as follow: $SH(M,|\chi|)=CH_{(|\chi|-1)}+SH(M-2^{|\chi|-1},k)$, where $CH_{(|\chi|-1)}$ is a SHG over $\chi-\{\sigma_n\}$, $k=\left|\log_2(M-2^{|\chi|-1})\right|$. Let RI_1, RI_2 denote respectively the nodes in $CH_{(|\chi|-1)}$ and $SH(M-2^{|\chi|-1},k)$, there is a link between the two nodes if and only if RI_1 and RI_2 are adjacent.

The following observation is drawn from the definition 4. All CHGs in a SHG are ordered according to the ascending order of their subscripts, then they constitute a hierarchical structure.

Definition 3.5. Given an arbitrary $SH(M,|\chi|)$, it can be denoted by a n-bit vector $V(SH(M,|\chi|))=(1,b_{n-2},\ldots,b_1,b_0)$, where b_i is a binary number, where $b_i=1$ iif $Ch_i(0\le i\le n-2)$ is in $SH(M,|\chi|)$, otherwise $b_i=0$.

Definition 3.6. Let $SS(Ch_i)= RI'(>i)\circ\{RI(<i)\}$, where $RI'(>i)$ is a RI, and denotes the multiplication for all elements whose subscript is $j(b_j=1\wedge j>i)$. $SS(Ch_i)$ is called subspace about Ch_i.

Some notations concerning the RI′ are as follows. $RI'(\le i,>j)$ is a RI, and denotes the multiplication for all elements which subscript is $q(b_j=1\wedge q>j\wedge q\le i)$. $SS(Ch_i)$ is called subspace about Ch_i. Similarly, $RI'(<i,>j)$ can be defined.

In SHG, the link between the node RIs A and B is labeled as $\lambda(A, B)$. $\lambda(A, B)$ is the i-th link if and only if A and B are the i-th adjacent RIs. The i-th link is denoted as λ_i.

3.2 SHG Construction

We use servers equipped with two network ports and switches to construct SHG architecture. In SHG, all switches are used as nodes, all servers are considered as links. A server is connected to two switches via communication links, which are assumed to be bidirectional. In addition, a link in SHG can express only a communication link, namely, not including a server.

The following example illustrates how a SHG is constructed. Firstly, we construct CH_4, CH_3 and CH_1, which are the building block to construct larger SHGs. Secondly, SH(26,5) is constructed from CH_4, CH_3 and CH_1. In SH(26,5), each $CH_i(i=1,3,4)$ is connected to all the other $CH_j(j=1,3,4$, and $i<>j)$ with one link. CH_i connects the other two CH_j as follows. Assign each server a 2-tuple (RI_i, RI_j), where RI_i and RI_j are adjacent

nodes. Then two switches with IDs RI_i and RI_j are connected with the server. For simplicity, if $|RI_i|>|RI_j|$, the server also can labeled as RI_i, else labeled as RI_j. The linking result for SH(26,5) is shown in Fig. 1. The solid-line connects two nodes which belong to the same CH, while the dotted-line connects the two nodes which belong to the two different CHs. For an arbitrary switch node, its name is a RI. As can be seen from the figure, CH_1, CH_3 and CH_4 constitute a hierarchical structure.

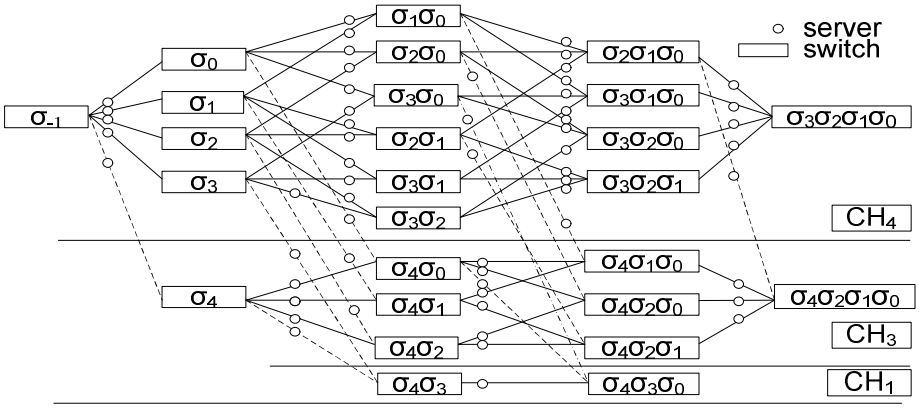

Fig. 1. SH(26,5) network

4 Routing Algorithm in SHG

4.1 Shortest Path Routing

In SHG, routing between any two servers is equivalent to routing between the corresponding switches. For simplicity, we only involve the switches. Switch is called as node in the following routing algorithms and paragraphs. SHG uses a simple and efficient single-path routing algorithm for unicast by exploiting the hierarchical structure of SHG.

Give node RIs A and B in SH(M,|χ|), let Tag(A,B)=(A-B)◦(B-A), and the shortest distance between them denoted as Δ(A,B)=|A-B|+|B-A|. The shortest path routing algorithm, called SPRouting, is shown in Fig. 2. Consider the source node identifier A and the destination node identifier B. When computing the path from A to B in a SHG, we first call the function search to calculate σi whose subscript is the minimum value in all subscripts of the tag(A, B)'s characters, and return the first link that interconnects A and σi◦A. Then, through an iterative process, we can calculate the sub-path from σi◦ A to B. The final path of SPRouting is the combination of link (A, σi◦ A) and the sub-path (σi◦ A, B).

```
Input: packet μ, source node A, and target node B
Onput: the server node B outputs the packet μ
1. If tag(A, B)=Ø
2.   {Send μ to the local node;}
3. Else
   {//Let i be the minimum value in all subscripts of the
        tag(A, B)'s characters, and λi exists.
4.      i=Search(A, B);
5.      Send μ to the next node over link λi;
6.      tag(A, B)=tag(A, B)-σi;//delete σi in tag
   }
```

Fig. 2. SPRouting algorithm

A message carries with it a tag and is sent through λ_i link only if the subscript i is the minimum value in all subscripts of the tag and the λ_i link exists. The algorithm checks the tag rightwards, starting from the current minimum subscript. The following take again the Fig. 1 as examples, in order to illustrate SPRouting algorithm. In order to facilitate our discussion, as shown in Table 1, we classify the positions of the source and the destination nodes into the following classes.

(1) A and B nodes belong to the same CH.
(2) A and B nodes don't belong to the same CH, and all λ_i exist.
(3) A and B nodes don't belong to the same CH, and there is one link λ_i at least that doesn't exist.

Table 1. Discussion on SPRouting algorithm

Class	Tag	Discussion
Class 1: $A=\sigma_1CH_4$, $B=\sigma_3\sigma_2\sigma_1CH_4$	$\{\sigma_3,\sigma_2\}$	A message is sent to $\sigma_2\sigma_1$ through λ_2 link on σ_1.
Class 1	$\{\sigma_3\}$	A message is sent to $\sigma_3\sigma_2\sigma_1$ through λ_3 link on $\sigma_2\sigma_1$
Class 2: $A=\sigma_3\sigma_2\sigma_1CH_4$, $B=\sigma_4\sigma_0CH_3$	$\{\sigma_4,\sigma_3,\sigma_2,\sigma_1,\sigma_0\}$	A message is sent to $\sigma_3\sigma_2\sigma_1\sigma_0$ through λ_2 link on $\sigma_3\sigma_2\sigma_1$
Class 2	$\{\sigma_4,\sigma_3,\sigma_2,\sigma_1\}$	A message is sent to $\sigma_3\sigma_2\sigma_0$ through λ_1 link on $\sigma_3\sigma_2\sigma_1\sigma_0$.
Class 2	$\{\sigma_4,\sigma_3,\sigma_2\}$	A message is sent to $\sigma_3\sigma_0$ through λ_2 link on $\sigma_3\sigma_2\sigma_0$.
Class 2	$\{\sigma_4,\sigma_3\}$	A message is sent to σ_0 through λ_3 link on $\sigma_3\sigma_0$.
Class 2	$\{\sigma_4\}$	A message is sent to $\sigma_4\sigma_0$ through λ_4 link on σ_0.
Class 3: $A=\sigma_4\sigma_3\sigma_0CH_1$, $B=\sigma_4\sigma_2\sigma_0CH_3$	$\{\sigma_3,\sigma_2\}$	On the node $\sigma_4\sigma_3\sigma_0$, there are three edges λ_0, λ_3, λ_4, and no edge λ_2. So, a message cannot be sent through λ_2 link, but only sent to $\sigma_4\sigma_0$ through λ_3 link on $\sigma_4\sigma_3\sigma_0$.
Class 3	$\{\sigma_2\}$	A message is sent to $\sigma_4\sigma_2\sigma_0$ through λ_2 link on $\sigma_4\sigma_0$.

In SPRouting algorithm, the number of iterations depends on the shortest distance between A and B. Time complexity of the algorithm is $O(|\chi|)$.

4.2 Parallel Paths

Two or more parallel paths between a source server and a destination server exist if they are node-disjoint, i.e., the intermediate servers and switches on one path do not appear on the other. The following theorem shows how to generate parallel paths between two switches.

Theorem 4.1. Consider any two node RIs $A,B \in CH_i$, and $|Tag(A,B)|=h\leq i$. Between A and B, there are i parallel paths whose length are less than or equal to h+2.

Proof. Let CH_i be over χ_i, where $|\chi|=i$. And $Tag(A,B)=\{\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h}\}$.

(1) Because Ch_i is a CHG, all links $\lambda_m(0\leq m\leq i)$ exist in Ch_i. We use $(\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h})$ to indicate a path from A to B. After a message carries with it a tag and is sent through $\lambda_{k_j}(1\leq j\leq h)$ link, σ_{k_j} is deleted from the tag. Then, either there exist the character σ_{k_j} in the address of both A and B, or not. This process can be expressed as a $\lambda_{k_j}(1\leq j\leq h)$ shift on $(\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h})$. So, there are h parallel paths whose length is h, namely shortest paths: $\{\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h}\}^h$. Where $\{\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h}\}^h$ denoted as h shifts on $(\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h})$.

(2) Let χ_i-Tag(A,B)=$\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}$. Where p+h=$|\chi_i|$,
$\{\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h}\}\cap\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}=\{\}$, and $\{\sigma_{k_1},\sigma_{k_2},\cdots,\sigma_{k_h}\}\cup\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}=\chi_i$.

To clarify, we use $(\sigma_{k_1}|\sigma_{n_j})(\sigma_{k_2},\cdots,\sigma_{k_h})(\sigma_{n_j}|\sigma_{k_1})$ to indicate $(\sigma_{n_j},\sigma_{k_2},\cdots,\sigma_{k_h},\sigma_{k_1})$, where $(\sigma_{n_j},\sigma_{k_2},\cdots,\sigma_{k_h},\sigma_{k_1})$ is a path from A to B. On this path, firstly, a message is sent through λ_{n_j} link. And then, the message is sent to along $(\sigma_{k_2},\cdots,\sigma_{k_h})$. Finally, the message is sent to B through λ_{n_j} link. The length of this path is h+2.Similarly, from A to B, all the paths whose length equal to h+2 are described as follows.

$$(\sigma_{k_1}|\sigma_{n_j})(\sigma_{k_2},\sigma_{k_3}\cdots,\sigma_{k_h})(\sigma_{n_j}|\sigma_{k_1}),\sigma_{n_j}\in\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}$$
$$(\sigma_{k_q}|\sigma_{n_j})(\sigma_{k_1},...,\sigma_{k_q\ 1},\sigma_{k_{q+1}},\cdots,\sigma_{k_h})(\sigma_{n_j}|\sigma_{k_q}),\sigma_{n_j}\in\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}$$
$$(\sigma_{k_h}|\sigma_{n_j})(\sigma_{k_2},\cdots,\sigma_{k_h\ 1})(\sigma_{n_j}|\sigma_{k_h}),\sigma_{n_j}\in\{\sigma_{n_1},\sigma_{n_2},\cdots,\sigma_{n_p}\}$$

So, there are i-h paths whose length equal to h+2.

According to the above (1) and (2), we can come to the conclusion.

Definition 4.1. Give Ch_i, Ch_j ($j<i$), the pivot set between Ch_i and Ch_j is defined as follows: $P_{i,j}$={A| A$\in$ $SS(Ch_i)$ and $\exists B\in CH_j$, A is adjacent to B}. $\forall x\in P_{i,j}$ is called the pivot.

Theorem 4.2. Give Ch_i, Ch_j ($j<i$). There are only 2^j links λ_i between the pivots in $P_{i,j}$ and the nodes in Ch_j.

Proof. (1) Prove the existence.

$$SS(Ch_i)=RI'(>i)\circ\{RI(<i)\}=RI'(>i)\circ\{RI(<i,\geq j)\}\circ\{RI(<j)\} \quad (1)$$

$$SS(Ch_j)=RI'(>j)\circ\{RI(<j)\}=RI'(>i)\circ RI'(\leq i,>j)\circ\{RI(<j)\}=RI'(>i)\circ\sigma_i\circ RI'(<i,>j)\circ\{RI(<j)\} \quad (2)$$

Let $\{RI(<i,\geq j)\}$ in Eq.1 be $M'(<i,>j)$ in Eq.2. Then, there are 2^j links λ_i.

(2) Prove the uniqueness by reductio.

Assume that A and B are adjacent. Then there exists $\lambda_q(q\neq i)$ which is a link between node $A\in SS(Ch_i)$ and node $B\in SS(Ch_j)$.

$$\text{For A, B, then A-B}=\sigma_q \quad (3)$$

For any node $C\in SS(Ch_i)$, and $D\in SS(Ch_j)$,

$$\text{then } \exists\sigma_i \quad \chi(SS(Ch_i))\wedge\sigma_i\in\chi(SS(Ch_j)), \text{ and } \sigma_i\in\chi(C\quad D). \quad (4)$$

According to Eq.3 and Eq.4, There are at least two different characters between A and B, such as σ_q and σ_i. So, A and B can't be adjacent.

4.3 Fault-tolerant Routing

Link failure between a source and a target server is defined as follows: the source and the destination servers are valid, but the source server is unable to communicate with the target server. The basic idea of fault-tolerant routing is to find an alternative link for the failure link, and complete the routing process between

the source and target servers. We take source routing approach to achieve fault-tolerance routing. The source server establishes a set of parallel paths to the destination server, and probes packet path on the set. When a path corresponding to packets fails, the source server selects an alternative available path from the detected paths, and continues packet transfer process to realize fault-tolerant routing. The fault-tolerant routing algorithm, called FTRouting, is shown in Fig. 3.

```
Input: source node A, target node B, and V(SH(M,|χ|))
Onput: A path between A and B
1. Path=Ø; CHS=Ø;
2. If i=j
      {//According to theorem 1, SearchPath generates a path which
ensures reliable transmission
3.     Path=SearchPath(A, B);
4.     Return
      }
5. Else
      {//Calculate the sum of the elements in V
6.     q=Add(V(SH(M,|χ|)));
7.     For r=0 To q Do
8.        For p=1 to C_q^r  Do
            {//Choose CH from IH to form a set which had not been
selected and the number of elements is p.
9.           CHS=Select(q, IH);
             //Given CHS = {CH_{l1},CH_{l2},⋯CH_{lp}},l1 < l2 < ⋯ < lp
10.          If  x_{l1} ∈P_{l1,l2}  and ... and  x_{lp} ∈P_{lp-1,lp}  and
                  Path(x_{l1},⋯,x_{lp}) is valid
11.            {Path=Path+Path(x_{i1},⋯,x_{ip}) ;
12.             Path=Path+SearchPath(A, x_{l1} )+SearchPath( x_{lp} ,B);
13.             Return;
               }
14.            Else exit;
             }
      }
```

Fig. 3. FTRouting algorithm

Consider SH(M,|χ|)(V,E). When computing the path from A to B in a SHG through FTRouting algorithm, we determine whether A and B belong to the same CGH. If so, we calculate a path between A and B according to the theorem 1. If not, we first calculate the number of CHs in the SHG, and assign to q. Then follow the steps below to search a path.

(1) Choose CHs from SH, the number of elements in these CHs is p. these CHs are denoted as $CHS = \{CH_{l_1}, CH_{l_2}, \cdots CH_{l_p}\}, l_1 < l_2 < \cdots < l_p$.

(2) If there exist $x_{l_1} \in P_{l_1,l_2}$, $x_{l_p} \in P_{l_p\ 1,l_p}$, and a $Path(x_{i_1},\cdots,x_{i_p})$, then the calculated path is as follows. Path(A, x_{l_1}), $Path(x_{l_1},\cdots,x_{l_p})$, Path($x_{l_p}$, B). Otherwise, return (1), and continue to the next round of FTRouting.

In FTRouting algorithm, the number of iterations depends on the number of iterations depends on the number of intermediate routing layers. Time complexity of the algorithm is $O(|\chi|^2)$.

5 Validation and Evalution

The metric aggregate bottleneck throughput (ABT) is defined as the throughput of the bottleneck flow times the total number of flows in the all-to-all traffic model. The metric mean latency (ML) is defined as the

number of cycles spent by a typical message from its source to its destination in packet-switching, taking the queuing delay into consideration, and throughput (TP) is the probability of a node receiving a message during a cycle, it indicates accepted traffic, or equivalently, the load. ABT and ML are two aspects of DCN's performance. They play important parts in the evaluation of system performance. In the following experiments, we validate SHG by simulation, Java language is used to construct SHG and Fat-Tree DCN simulation platforms.

(1) Experiment 1. We compare SHG with Fat-Tree. We assume that such network structures interconnect the same number of servers with the same type of switches. and the network traffic across all links is 1Gbps. By simulating the all-to-all traffic within the data center, we test ABT trends under server failure rate. We kept the server scale fixed to 256 and 1024, respectively. For SHG, Given SH(97,6), SH(384,9). Then 256 links and 1024 links are randomly selected from SH(97,6) and SH(384,9), respectively. All the selected links are used to place the servers. Fig. 4 shows the results w.r.t. an increasing server failure rate. In the legend, N and TF are denoted as the number of servers and Fat-Tree, respectively.

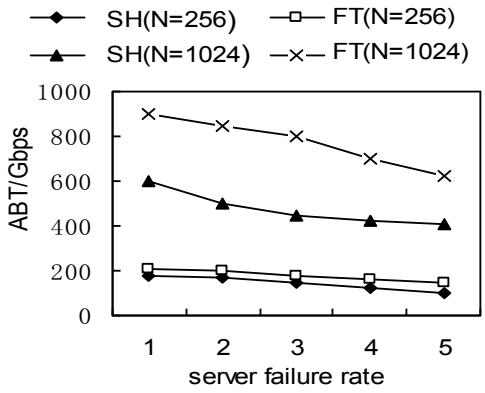

Fig. 4. FTRouting algorithm

By Fig. 4, it is clear that two kinds of structure show a relatively smooth downward trend with the rise of the server failure rate. The declining trends in ABT of SHG structure become more gradual, indicating that the impact of the expansion of SHG scale on performance is small. In the same network scale, the ABT of SGH is lower than that of Fat-Tree. Moreover, the larger the scale of DCN, the larger the ABT gap between SGH and Fat-Tree. This shows that the hierarchical structure of SGH makes the network traffic more evenly, thereby reducing the probability of occurrence of bottleneck flow.

(2) Experiment 2. Fig. 5 depicts ML vs. TP for three SHG of SH(536, 10)={CH_9, CH_4, CH_3}, SH(602, 10)={CH_9, CH_6, CH_4, CH_3, CH_1} and SH(794, 10)={CH_9, CH_8, CH_4, CH_3, CH_1}.

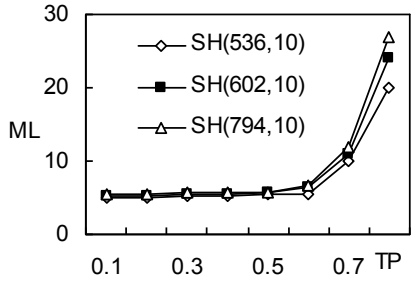

Fig. 5. ML vs. TP for SH(536,10), SH(602,10), and SH(794,10)

For the three SHGs, ML grows slowly until T reaches 0.6, and starts to change quickly thereafter. The results imply that SHG can ensure good scalability.

6 Conclusions

In this paper, we propose SHG, a scalable hierarchical approach, to provide the topology of architecting DCN. SHG is defined based on CHG. We design and implement the shortest path routing and the fault-tolerant routing algorithms in SHG, and derive conclusions from the simulation experiments. The results show that SHG can easily be constructed, while ensuring good scalability. Automatic configuration can cut labour costs, and reduce the risk of errors in configuring. In our future work, we will study how to automatically configure SHG, and propose a method for automatically configuring address with high reliability, low cost, and ease of use. This work was sponsored by Qing Lan project (Jiangsu province, China) and open fund project of Jiangsu provincial research and development center of intelligent sensor network engineering technology, China (ZK13-02-03, Software technology, platform and application for sensor network).

Competing Interests

Author has declared that no competing interests exist.

References

[1] Al-Fares M, Loukissas A, Vahdat A. A scalable, commodity data center network architecture. ACM SIGCOMM Computer Communication Review. 2008;38(4):63-74.

[2] Mysore RN, Pamboris A, Farrington N, Huang N, Miri P, Radhakrishnan S, et al. Port Land: A scalable fault-tolerant layer 2 data center network fabric. SIGCOMM. 2009;9:39-50.

[3] Heller B, Seetharaman S, Mahadevan P, Yiakoumis Y, Sharma P, Banerjee S, et al. ElasticTree: Saving Energy in Data Center Networks. NSDI. 2010;10:249-264.

[4] Greenberg A, Hamilton JR, Jain N, Kandula S, Kim C, Lahiri P, et al. VL2: A scalable and flexible data center network. Communications of the ACM. 2011;54(3):95-104.

[5] Farrington N, Porter G, Radhakrishnan S, Bazzaz HH, Subramanya V, Fainman Y, et al. Helios: A hybrid electrical/optical switch architecture for modular data centers. ACM SIGCOMM Computer Communication Review. 2011;41(4):339-350.

[6] Wang G, Andersen DG, Kaminsky M, Papagiannaki K, Eugene Ng TS, Kozuch M, et al. c-Through: Part-time optics in data centers. ACM SIGCOMM Computer Communication Review. 2011;41(4):327-338.

[7] Chen K, Singla A, Singh A, Ramachandran K, Xu L, Zhang Y, et al. OSA: An optical switching architecture for data center networks with unprecedented flexibility. IEEE/ACM Transactions on Networking (TON). 2014;22(2):498-511.

[8] Halperin D, Kandula S, Padhye J, Bahl P, Wetherall D. Augmenting data center networks with multi-gigabit wireless links. ACM SIGCOMM Computer Communication Review. ACM. 2011;41(4):38-49.

[9] Vardhan H, Ryu SR, Banerjee B, Prakash R. 60 ghz wireless links in data center networks. Computer Networks. 2014;58:192-205.

[10] Guo D, Chen T, Li D, Liu Y, Liu X, Chen G. BCN: Expansible network structures for data centers using hierarchical compound graphs. INFOCOM, 2011 Proceedings IEEE. 2011;61-65.

[11] Wu H, Lu G, Li D, Guo C, Zhang Y. MDCube: A high performance network structure for modular data center interconnection. Proceedings of the 5th international conference on Emerging networking experiments and technologies. ACM. 2009;25-36.

[12] Huang L, Jia Q, Wang X, Yang S, Li B. Pcube: Improving power efficiency in data center networks[C]//Cloud Computing (CLOUD), 2011. IEEE International Conference. 2011;65-72.

[13] Kliazovich D, Bouvry P, Khan SU. DENS: Data center energy-efficient network-aware scheduling. Cluster computing. 2013;16(1):65-75.

[14] Perelló J, Spadaro S, Ricciardi S, Careglio D, Peng S, Nejabati R, et al. All-optical packet/circuit switching-based data center network for enhanced scalability, latency and throughput. Network, IEEE. 2013;27(6):14-22.

[15] Ghosh A, Ha S, Crabbe E, et al. Scalable multi-class traffic management in data center backbone networks. Selected Areas in Communications. IEEE. 2013;31(12):2673-2684.

[16] Koutitas G, Tassiulas L. Smart grid technologies for future radio and data center networks. Communications Magazine. IEEE. 2014;52(4):120-128.

[17] Singla A, Godfrey PB, Kolla A. High throughput data center topology design. Proceedings of the 11th USENIX conference on networked systems design and implementation. USENIX Association, Berkeley. 2014;29-41.

[18] Dong B. architecting large scale wireless sensor networks publish/subscribe applications: A Graph-oriented approach. Applied Mechanics and Materials. 2013;321:2768-2771.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
www.sciencedomain.org/review-history.php?iid=1142&id=6&aid=9216