CLOT: A Cost-effective Low-latency Overlaid Torus-based Network Architecture for Data Centers

Ting Wang*, Zhiyang Su*, Yu Xia*, Mounir Hamdi*‡
Hong Kong University of Science and Technology*, Hamad Bin Khalifa University‡
{twangah, zsuab, rainsia, hamdi}@cse.ust.hk

***Abstract*—In this paper, we present the design, analysis, and implementation of a novel data center network architecture named *CLOT*, which delivers significant reduction in the network diameter, network latency, and infrastructure cost. *CLOT* is built based on a switchless torus topology by adding only a number of most beneficial low-end switches in a proper way. Forming the servers in close proximity of each other in torus topology well implements the network locality. The extra layer of switches largely shortens the average routing path length of torus network, which increases the communication efficiency. We show that *CLOT* can achieve lower latency, smaller routing path length, higher bisection bandwidth and throughput, and better fault tolerance compared to both conventional hierarchical data center networks as well as the recently proposed CamCube network. Coupled with the coordinate based translated IP addresses, the carefully designed POW routing algorithm helps *CLOT* achieve its maximum theoretical performance. The sufficient mathematical analysis and theoretical derivation prove both guaranteed and ideal performance of *CLOT*.**

***Keywords*—Data Center Network, Torus, Network Topology**

I. Introduction

Serving as the core infrastructure for the cloud providers as well as large-scale enterprise applications, the data center network (DCN) plays a key role in determining the performance of service provided to users. The traditional hierarchical switched DCN topology, besides being very costly, suffers high oversubscription ratio towards higher layers leading to serious communication bottleneck. Thus, researchers proposed several more cost-effective DCN architectures such as BCube [8], DCell [7], SprintNet [13], CamCube [3], Small-World [10] and NovaCube [12], where these server-centric architectures abandon expensive high-end network switches by using only low-end switches or even no switches at all. In addition, the forwarding functionality is shifted to the servers, which helps achieve higher bisection bandwidth and better fault tolerance with richer connections. However, they suffer high latencies due to their relatively long routing path length. As a result, some optical DCN topologies like OSA [4] have been proposed. Compared with packet switching, the optical circuit switching can provide higher bandwidth and lower latency in transmission. However, the optics suffer from slow switching speed which can take as long as tens of milliseconds, and cannot achieve full bisection bandwidth at packet granularity.

Based on these observations, in this paper we propose a novel DCN architecture named *CLOT*, which is built on a k-ary n-D torus topology whose various unique advantages have been carefully exploited in [12]. Based on torus *CLOT* well implements the network locality forming the servers in close proximity to each other, which increases the communication efficiency. Besides, in response to the serious issue of long routing path length in torus topology, *CLOT* employs a number of low-end switches connecting the most distant node-pairs in each dimension, which largely shortens both network diameter and average path length; this in turn reduces the network latency. Meanwhile, *CLOT* also largely improves the bisection bandwidth, throughput and fault tolerance with better path diversity. Moreover, from the perspective of cost-effectiveness, *CLOT* also far outperforms other hierarchical topologies like FatTree [2] with regard to the network cost. Furthermore, the geographical address assignment mechanism enables content routing such as key-value stores in addition to traditional routing, and works well with legacy TCP/IP protocol.

The rest of the paper is organized as follows. First we briefly review related works in Section II. Then Section III presents the design and analysis of *CLOT*. Afterwards, Section IV discusses the network abstraction layers and the network address translation mechanism. The routing scheme and flow control designs are shown in Section V. Finally, Section VI concludes this paper with some of our future work plans.

II. Related Work

As a low cost, efficient and robust network architecture, torus fabric has received considerable research interest in recent years. Besides being widely used in the High Performance Computing (HPC), the torus-based interconnects have also recently been introduced to data center clusters, and the typical examples include CamCube [3], Small-World [10].

CamCube [3] is a shipping container sized DCN based on a 3D torus topology, and it is the first work that introduces the torus fabric into data center interconnects. CamCube is one prototype that utilizes a low-level link oriented API to allow applications to implement their own particular routing protocols to optimize the application-level performance. These customized routing protocols run as services in the servers, which is similar to overlay network. The critical drawback is the long routing path length of torus based network, which results in low routing efficiency and high latency.

In order to decrease the average routing path length, researchers proposed a random data center topology named Small-World (SWDC) [10] which is built upon a regular pattern, such as ring, torus or tube. Small-World reduces the average routing path length by introducing a number of random links. The degree of each server is limited to six, where the number of random links of each node in SW-Ring, SW-2DTrous, and SW-3DHexTorus are 4, 2, and 1, respectively.

978-1-4673-6432-4/15/$31.00 ©2015 IEEE

However, the deterministic short-path based greedy routing algorithm results in low worst-case throughput and poor load balancing, which may lead to network congestion.

III. *CLOT* Network Structure

This section presents the design principle of *CLOT* architecture with comprehensive performance analysis from various aspects. Prior to introducing the physical interconnections, we first bring forward several theorems with proof, which provides the theoretical basis of *CLOT* design.

Theorem 3.1. *For any node A(a_1, a_2, ..., a_n) in a k-ary n-D torus (when k is even), assuming $\widehat{F}_A = B(b_1, b_2, ..., b_n)$ is the farthest node from A, then B is unique and B's unique farthest node is exactly A.*

Proof. In a k-ary n-D torus, if B(b_1, b_2, ..., b_n) is the farthest node from A(a_1, a_2, ..., a_n), where $a_i \in [0, k)$, $b_i \in [0, k)$, then:

$$b_i = (a_i + \frac{k}{2}) \ mod \ k, \quad for \ \forall i \in [1, n] \tag{1}$$

Since the result of $(a_i + \frac{k}{2}) \ mod \ k$ is unique, thus $\forall b_i \in [0, k)$ is unique. Hence, A's farthest node B is unique. Next, assume $\widehat{F}_B$=A′(a'_1, a'_2, ..., a'_n), similarly we have:

$$a'_i = (b_i + \frac{k}{2}) \ mod \ k, \quad for \ \forall i \in [1, n] \tag{2}$$

By combining (1) and (2), we can get:

$$a'_i = \{[(a_i + \frac{k}{2}) \ mod \ k] + \frac{k}{2}\} \ mod \ k \tag{3}$$

$$\because a_i \in [0, k), \quad \therefore a_i + \frac{k}{2} \in [\frac{k}{2}, k + \frac{k}{2}) \tag{4}$$

(1) For the case of $a_i + \frac{k}{2} \in [\frac{k}{2}, k)$, we have

$$\begin{aligned} a'_i &= \{[(a_i + \frac{k}{2}) \ mod \ k] + \frac{k}{2}\} \ mod \ k \\ &= (a_i + \frac{k}{2} + \frac{k}{2}) \ mod \ k = (a_i + k) \ mod \ k = a_i \end{aligned} \tag{5}$$

(2) For the case of $a_i + \frac{k}{2} \in [k, k + \frac{k}{2})$, we have

$$\begin{aligned} a'_i &= \{[(a_i + \frac{k}{2}) \ mod \ k] + \frac{k}{2}\} \ mod \ k \\ &= (a_i + \frac{k}{2} - k + \frac{k}{2}) \ mod \ k = a_i \ mod \ k = a_i \end{aligned} \tag{6}$$

As a consequence of the above, $a'_i = a_i$ for $\forall i \in [1, n]$. Therefore, A′(a'_1, a'_2, ..., a'_n) = A(a_1, a_2, ..., a_n), which means the farthest node from B is exactly A. This ends the proof. □

Theorem 3.2. *In an n-D Torus with radix k, for any node A($a_1, a_2, ..., a_n$), its farthest node $\widehat{F}_A$ = (($a_1 + \lfloor \frac{k}{2} \rfloor$) mod k, ($a_2 + \lfloor \frac{k}{2} \rfloor$) mod k, ..., ($a_n + \lfloor \frac{k}{2} \rfloor$) mod k).*

Proof. This theorem is simple and obvious. For any node A($a_1, a_2, ..., a_n$) in k-ary n-cube, its farthest nodes on the first dimension are (($a_1 + \lfloor \frac{k}{2} \rfloor$) mod k, 0, ..., 0), ..., (0, ..., ($a_i + \lfloor \frac{k}{2} \rfloor$) mod k, 0, ..., 0), ..., (0, ..., 0, ($a_n + \lfloor \frac{k}{2} \rfloor$) mod k). By analogy, A′s farthest nodes on subsequent higher dimensions can be done in the same manner. For instance, its farthest nodes on the i-th dimension are to set i number of coordinates to be ($a_j + \lfloor \frac{k}{2} \rfloor$) mod k, where $j \in [1, n]$, and the rest are set to be 0. Consequently, there are $\binom{n}{i}$ number of farthest nodes on the i-th dimension. Therefore, A′s farthest node in an n-D Torus with radix k is $\widehat{F}_A$ = (($a_1 + \lfloor \frac{k}{2} \rfloor$) mod k, ($a_2 + \lfloor \frac{k}{2} \rfloor$) mod k, ..., ($a_n + \lfloor \frac{k}{2} \rfloor$) mod k). This ends the proof. □

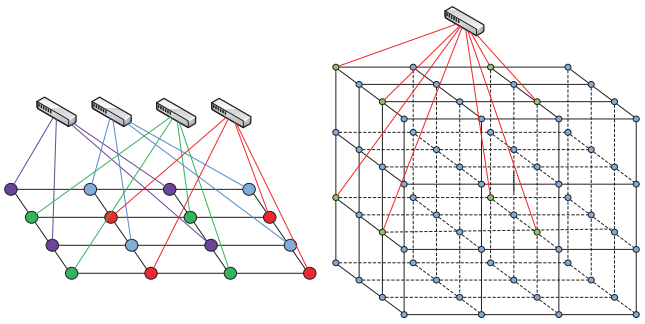

Fig. 1. Examples of 2D&3D *CLOT* topology (without wrap around links).

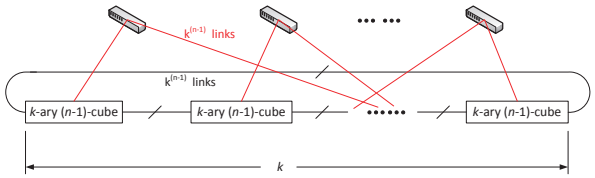

Fig. 2. An example of n-D *CLOT* with radix k.

Theorem 3.3. *In a k-ary n-D Torus, for a certain node A, denoting its farthest nodes on the i-th dimension as $\{\widehat{F}_A^i\}$, then the total number of A's farthest nodes in all dimensions is $N = \sum_1^n |\{\widehat{F}_A^i\}| = 2^n - 1$.*

Proof. Derived from Theorem 3.2, we can get $|\{\widehat{F}_A^i\}| = \binom{n}{i}$. Thus, the total number of farthest nodes of node A in all dimensions can easily be computed as $N = \sum_1^n |\{\widehat{F}_A^i\}| = \sum_1^n \binom{n}{i} = 2^n - 1$, which ends the proof. □

A. Physical Structure

1) Key Principle: As aforementioned, Torus suffers a lot from its long network diameter, which results in communication inefficiency and extra latency. The key principle of *CLOT* construction is to reduce the routing path length and improve its overall network performance while retaining the Torus's merits. Based on regular torus topology, *CLOT* is established by adding a number of most beneficial small low-end switches. More specifically, in *CLOT* each node and its $2^n - 1$ farthest nodes in different dimensions are connected via a low-end switch. In order to better illustrate the specific architecture of *CLOT*, some representative cases are provided as below.

2) Cases:

- **Case #1:** Take 2-D *CLOT* for example, for any node A, as indicated in Theorem 3.3, the number of its farthest nodes is three ($2^2 - 1$) in total. Thus, each switch needs to connect any node and its three farthest nodes in total, where a 4-port low end switch is needed. The four nodes under the same switch are: (a_i, a_j), (($a_i + \lfloor \frac{k}{2} \rfloor$) mod k, a_j), (a_i, ($a_j + \lfloor \frac{k}{2} \rfloor$) mod k), and (($a_i + \lfloor \frac{k}{2} \rfloor$) mod k,

$(a_j + \lfloor \frac{k}{2} \rfloor)$ mod k), respectively, where $i, j \in [1, k]$, and $a_i, a_j \in [0, k-1]$. Fig.1 (left) illustrates the architecture of a 2D *CLOT*.

- **Case #2:** Likewise, for the case of 3D *CLOT*, eight-port switches are needed. The eight nodes under the same switch are: (a_i, a_j, a_z), $((a_i + \lfloor \frac{k}{2} \rfloor)$ mod k, a_j, $a_z)$, $(a_i$, $(a_j + \lfloor \frac{k}{2} \rfloor)$ mod k, $a_z)$, $(a_i$, a_j, $(a_z + \lfloor \frac{k}{2} \rfloor)$ mod $k)$, $((a_i + \lfloor \frac{k}{2} \rfloor)$ mod k, $(a_j + \lfloor \frac{k}{2} \rfloor)$ mod k, $a_z)$, $((a_i + \lfloor \frac{k}{2} \rfloor)$ mod k, a_j, $(a_z + \lfloor \frac{k}{2} \rfloor)$ mod $k)$, $(a_i$, $(a_j + \lfloor \frac{k}{2} \rfloor)$ mod k, $(a_z + \lfloor \frac{k}{2} \rfloor)$ mod $k)$, and $((a_i + \lfloor \frac{k}{2} \rfloor)$ mod k, $(a_j + \lfloor \frac{k}{2} \rfloor)$ mod k, $(a_z + \lfloor \frac{k}{2} \rfloor)$ mod $k)$, respectively, where $i, j, z \in [1, k]$, and $a_i, a_j, a_z \in [0, k-1]$. Fig.1 (right) gives an example of a 4*4*4 3D *CLOT*. For simplicity, the wrap around links are not displayed and only one switch is demonstrated.
- Generally, for a n-D *CLOT*, each switch connects to 2^n nodes: $((a_1 + i * \lfloor \frac{k}{2} \rfloor)$ mod k, $(a_2 + j * \lfloor \frac{k}{2} \rfloor)$ mod k, ..., $(a_n + r * \lfloor \frac{k}{2} \rfloor)$ mod $k)$, where $i, j, r \in \{0, 1\}$.

B. Key Properties & Performance Analysis

1) Network Diameter: The length of routing path in a network greatly impacts the communication efficiency and transmission latency, while the network diameter largely determines the routing path length. Thus, a lower network diameter should be offered as a basic feature of network design. From this aspect, *CLOT* is highlighted by its low network diameter.

Theorem 3.4. *For a n-D CLOT with radix k, its network diameter is* $D_{CLOT} = \left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor$.

Proof. As known, the network diameter of a regular k-ary n-Torus is $\lfloor \frac{k}{2} \rfloor n$, which means that starting from any node it can reach all the other nodes in the network within $\lfloor \frac{k}{2} \rfloor n$ hops. Assume S_i denotes the set of nodes that can be reached starting from A at the i-th hop, then the universal set of all nodes can be expressed as $S = \sum_{i=0}^{\lfloor \frac{k}{2} \rfloor n} S_i$. In *CLOT*, after connecting each node with its $2^n - 1$ farthest nodes in different dimensions through a switch, we define S'_i denotes the set of nodes that are i hops from node A, then there is: (1) when $\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor$ is even, we have $S'_0 = S_0$, $S'_1 = S_1$, $S'_2 = S_2 + \sum_{i=1}^{n} S_{i * \lfloor \frac{k}{2} \rfloor}$, ..., $S'_{\frac{\lfloor \frac{k}{2} \rfloor + 1}{2} * n - 1} = S_{\frac{\lfloor \frac{k}{2} \rfloor + 1}{2} * n - 1} + S_{\frac{\lfloor \frac{k}{2} \rfloor + 1}{2} * n}$. (2) when $\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor$ is odd, we have $S'_0 = S_0$, $S'_1 = S_1$, $S'_2 = S_2 + \sum_{i=1}^{n} S_{i * \lfloor \frac{k}{2} \rfloor}$, ..., $S'_{\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor - 1} = S_{\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor - 1} + S_{\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor + 1}$, $S'_{\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor} = S_{\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor}$. It can be concluded that in *CLOT* any node A can reach all the other nodes within $\frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n$-1 or $\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor$ hops. Therefore, the network diameter of *CLOT* is $\left\lfloor \frac{\lfloor \frac{k}{2} \rfloor + 1}{2} n \right\rfloor$. □

2) Bisection Bandwidth: If the network is segmented into two equally sized groups such that the bandwidth between these two groups is minimum, then bisection bandwidth is calculated as the sum of link capacities between the two halves. In a word, the bisection bandwidth represents the bandwidth across the "narrowest" part of the network. It can be used to evaluate the worst-case network capacity [9].

Theorem 3.5. *The bisection bandwidth of n-D CLOT with radix k is:* $B_C = k^n + 4k^{n-1}$.

Proof. Assume *CLOT* network $N(N_1, N_2)$ is partitioned into two equal disjoint halves N_1 and N_2, and the set of links connecting two parts N_1, N_2 is denoted as $L(N_1, N_2)$. Each element of $L(N_1, N_2)$ is a bidirectional link with one node in N_1 and the other node in N_2. Thus, there are $|L(N_1, N_2)|$ bidirectional links or $2|L(N_1, N_2)|$ unidirectional channels at the bisection position. If the bandwidth of each unidirectional channel is set to be 1, then the bisection bandwidth of *CLOT* will be $B_C = 2|L(N_1, N_2)|$. For a k-ary n-D *CLOT* as depicted in Fig.2, when k is even, there is even number of k k-ary $(n-1)$-D *CLOT*, which can be divided by the minimum bisection into two equal groups with $2k^{n-1}$ regular links and $\frac{k^n}{2}$ switch-server links. Therefore, there is $|L(N_1, N_2)| = 2k^{n-1} + \frac{k^n}{2}$. As a result, $B_C = 2|L(N_1, N_2)| = k^n + 4k^{n-1}$, which ends the proof. □

Compared with the traditional regular Torus network whose bisection bandwidth is $B_T = 4k^{n-1}$, *CLOT* effectively improves the bisection bandwidth by at least $\frac{B_C - B_T}{B_T} = \frac{k^n + 4k^{n-1} - 4k^{n-1}}{4k^{n-1}} = \frac{k}{4} \geq 25\%$, and the ratio increases accordingly as k increases. For instance, k=10 implies a 250% increasement to the bisection bandwidth.

3) Throughput: The network throughput is an important reference indicator to evaluate the network capacity of a topology. Throughput not only is limited by bisection bandwidth, but also largely depends on the traffic pattern, routing algorithm and flow control mechanism. However, we can evaluate the ideal throughput of a topology under the optimal routing and flow control. The maximum throughput occurs as some channel in the network is saturated and the network cannot carry more traffic. Assume the workload on a channel c is γ_c, and the channel bandwidth is b_c, then the maximum channel workload of the network is $\gamma_{max} = max\{\gamma_c, c \in C\}$. The ideal throughput Θ_{ideal} then can be obtained as:

$$\Theta_{ideal} = \frac{b_c}{\gamma_{max}} \tag{7}$$

Considering the uniform traffic pattern, the maximum channel load at the bisection channel has a lower bound which results in an upper bound on throughput. Because the *CLOT* is both node- and edge-symmetric, thus on average half of the k^n packets must cross through the B_C bisection channels. With assumption of the existence of optimal routing and flow control, the $\frac{k^n}{2}$ packets will be evenly distributed among all bisection channels which results in ideal throughput, and the lower bound of bisection channel load can be calculated as:

$$\gamma_{max} \geq \frac{\frac{k^n}{2}}{B_C} = \frac{\frac{k^n}{2}}{k^n + 4k^{n-1}} = \frac{k}{2k + 8} \tag{8}$$

Combining Equations 7 and 8, we can derive that

$$\Theta_{ideal} = \frac{b_c}{\gamma_{max}} \leq \frac{2k + 8}{k} b_c \tag{9}$$

Compared with the regular torus network whose ideal throughput is only $\frac{8b_c}{k}$ [6], *CLOT* exhibits a considerable improvement to the network capacity regarding to the maximum throughput. From another perspective, if the throughput under a certain routing algorithm is normalized to the network capacity $\widehat{\Theta}_n = \frac{\gamma_{max}}{\gamma_n(\widehat{R})}$, where $\widehat{R}$ denotes a routing algorithm, then in torus its maximum normalized throughput $\widehat{\Theta}_n$ can reach as high as 50% under a throughput optimal routing algorithm like Valiant routing (VAL) [11]. Consequently, in *CLOT* the maximum normalized worst-case throughput can be achieved by at least 62.5%, which which is further evidence of *CLOT*'s better performance.

4) Path Diversity: CLOT is vastly superior to other DCN architectures in path diversity. The number of distinct paths existing in *CLOT* is too huge to be calculated exactly, for simplicity, we first consider only the equal-cost shortest paths with the same direction in a regular torus without considering the switch-server links. Assume two nodes $A(a_1, a_2, ..., a_n)$ and $B(b_1, b_2, ..., b_n)$ where A and B are separated by $\Delta_i = |a_i - b_i|$ hops in the i-th dimension, then the total number of minimal paths N_{AB} between A and B is given by:

$$N_{AB} = \prod_{i=1}^{n} \binom{\sum_{j=i}^{n} \Delta_j}{\Delta_i} = \frac{(\sum_{i=1}^{n} \Delta_i)!}{\prod_{i=1}^{n} \Delta_i!} \tag{10}$$

where the term $\binom{\sum_{j=i}^{n} \Delta_j}{\Delta_i}$ computes the number of ways to choose where to take the Δ_i hops out of the remaining $\sum_{j=i}^{n} \Delta_j$ hops. For example, given n=3, Δ_x=4, Δ_y=5, Δ_z=6, the total number of minimal paths N_{AB} is as high as 630630, and it increases rapidly with dimension. Besides, if taking the switch-server links into consideration the number of possible paths will be much larger, and the number of non-minimal possible is nearly unlimited. The good path diversity of *CLOT* offers many benefits to the network. On the one hand, the traffic can be selectively distributed over these equal-cost paths which can help achieve good load balancing. On the other hand, this path diversity also provides good fault tolerance where the traffic can route around the faulty channels by taking alternative equal-cost paths.

TABLE I
THE PER-PORT COST ON DIFFERENT TYPES OF SWITCHES.

Number of Ports	Price per Port (US $)
16	150
16-32	450
>32	650

TABLE II
THE COMPARISON OF COST BETWEEN FAT-TREE AND 3-D *CLOT*.

Network Size	Switch Number (p-port switch)		Switch Cost (US $)	
	Fat-Tree	*CLOT*	Fat-Tree	*CLOT*
500	199 (p=13)	63 (p=8)	376,086	9,450
1,000	315 (p=16)	125 (p=8)	750,047	18,750
5,000	922 (p=27)	625 (p=8)	11,262,119	93,750
10,000	1463 (p=34)	1250 (p=8)	32,522,033	187,500
16,000	2000 (p=40)	2000 (p=8)	52,000,000	300,000
20,000	2321 (p=43)	2500 (p=8)	65,005,758	375,000
100,000	6787 (p=74)	12500 (p=8)	325,045,783	1,875,000
200,000	10773 (p=93)	25000 (p=8)	650,049,875	3,750,000

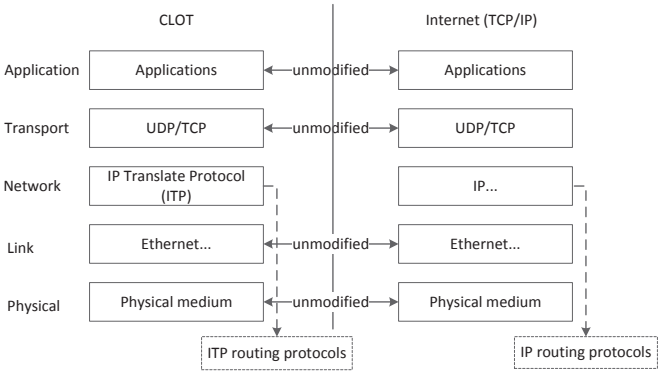

Fig. 3. The *CLOT* network abstraction layers.

5) Cost-effectiveness: Apart from its strong advantages in network performance, *CLOT* is also very cost-effective.

Theorem 3.6. *In a k-ary n-D CLOT, the total number of switches is $N_{switch} = (\frac{k}{2})^n$ with 2^n ports on each switch.*

Proof. In a n-D *CLOT* with radix k, it holds k^n nodes while each low-end switch connects 2^n nodes, thus the number of switches needed is calculated by $N_{switch} = (\frac{k}{2})^n$. □

As known, the per-port-price of a switch increases with its number of ports. Based on careful investigations and statistics on the Amazon [1] online quoted prices, Table I gives the average per-port-price for 10GE switches with different numbers of ports. Table II exhibits the comparison of switch cost between different sized 3D *CLOT*s and FatTree networks. It can be seen that a 3D *CLOT* uses only low-end 8-port cheap switches, while Fat-Tree requires high-end expensive switches with more ports. Although *CLOT* uses more switches than FatTree when the network size is larger than 16,000 servers, *CLOT* still yields a much lower switch cost. For example, FatTree requires an expenditure of 650,049,875$ on switches to scale up to 200,000 servers while *CLOT* only needs 375,000$ for the same network size which is around 173 times cheaper, which demonstrates the cost-effectiveness of *CLOT*.

IV. NETWORK LAYERING AND ADDRESS TRANSLATION

A. Network Layering

Similar to the Internet protocol suite, *CLOT* network also uses encapsulation to provide abstraction of protocols and services. As illustrated in Fig.3, the layered protocol stack in *CLOT* network is also divided into five layers (application, transport, network, link and physical), which enables an application to use a set of protocols to send its data down the layers, being further encapsulated at each level.

The major difference lies in the network layer, where the traditional Internet uses an IP address to locate different hosts while *CLOT* uses coordinates to direct the data transmission with the benefit of the perfect symmetry of topology, which facilitates the routing greatly. Coupled with the geographical identities, the geographical routing and addressing APIs make it possible for *CLOT* to implement applications like key-value stores, map-reduce operation and coordination protocols more efficiently. In order to keep the running applications unmodified, *CLOT* makes no changes to the transport protocols which

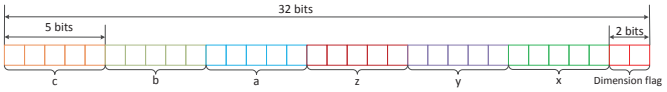

Fig. 4. The correspondence between IPv4 and *CLOT* coordinates.

enables the applications to directly adopt the functionality of TCP or UDP. Since the IP address must be provided when creating sockets for TCP and UDP, however, the *CLOT* only has coordinates, so the network needs an adaptation layer to translate coordinates to the IP address format.

B. Network Address Translation

In order to make the *CLOT* network compatible with the legacy TCP/IP protocol, we designed an address translation mechanism to implement the convention between IPv4 address and *CLOT* coordinates.

As illustrated in Fig.4, a 32-bit IPv4 address is divided into seven segments including six pieces with five bits and one piece with two bits. The coordinate of each dimension is denoted by the five-bit slice, and the remaining two bits are a dimension flag which is used to indicate the number of dimensions. In this way, a 32-bit IPv4 address can support up to six dimensions, where a 6-D *CLOT* can hold up to 2^{30}= 1,073,741,824 (1 billion) servers, thus this kind of division is reasonable and adequate even for a large scale data center. However, the two-bit dimension flag can not represent six dimensions. Here we define that only the dimension flag with "11" indicates a 6D network address. When the number of dimensions is less than six, the address space of last dimension will not be used. Therefore, when the dimension flag is "10", we make use of the first three bits of the last dimension's address space to represent the specific dimension. The rule of dimension correspondence is illustrated in Table III, and other values are currently considered illegal.

TABLE III
THE REPRESENTATION OF DIFFERENT SPECIFIC DIMENSIONS.

Dimension Flag (binary)	First 3 Bits of c (binary)	Dimension Number (decimal)
11	xxx	6
10	101	5
10	100	4
10	011	3
10	010	2
10	001	1

V. Routing Scheme and Flow Control

This section presents a Probabilistic Oblivious Weighted routing algorithm named POW and flow control mechanism for *CLOT*, which aim to achieve good load balancing with minimum routing path and high throughput.

A. POW Routing Algorithm

Different from the deterministic routing algorithms (e.g. DOR [5]) whose routing is directly determined by the source and destination address eliminating path diversities provided by torus topology incurring poor load balancing or even congestion, POW makes a probabilistic decision at each routing step according to a distance-based probability function. Here we use the notation Δ_{SD} to denote the DOR [5] distance (without switches) between node S and D.

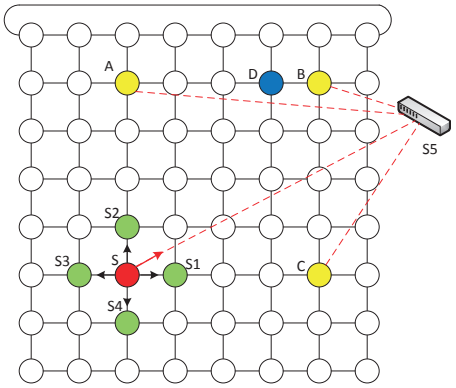

Fig. 5. POW routing in an 8*8 2-D *CLOT* (for simplicity not all switches and wrap-around links are shown).

Without loss of generality, for simplicity we use 2D *CLOT* to illustrate the working procedures of POW algorithm. As shown in Fig.5, assume a source node S needs to send data to a destination node D, then S has five directions in its first step to route the data which are S_1, S_2, S_3, S_4 and S_5, where S_5 is a switch node. In order to select the most beneficial next-hop node, each direction is assigned a probability based on the distances between each next-hop node and destination node. Then POW decides the next-hop node according to their probabilities. In order to guarantee the packet delivery, the distance Δ_{S_iD} between the chosen node S_i and destination D must be smaller than Δ_{SD}, i.e. $\Delta_{S_iD} < \Delta_{SD}$. The normalized probability function is computed as below:

$$p_i = \frac{\frac{1}{\Delta_i^2}}{\sum_{i=1}^{\psi} \frac{1}{\Delta_i^2}} \tag{11}$$

where ψ is the number of satisfied neighbours of the source node. For example, in Fig.5 we have Δ_{SD}=7, and the distances between S's neighbour nodes and destination D are Δ_{S_1D}=6, Δ_{S_2D}=6, Δ_{S_3D}=8, Δ_{S_4D}=6, Δ_{S_5D}=2, respectively, where S_3 fails to meet the requirement of $\Delta_{S_iD}<\Delta_{SD}$. Thus, the probability of choosing S_1 as the next-hop is $p_{S_1}=\frac{\frac{1}{\Delta_{S_1D}^2}}{\sum \frac{1}{\Delta_{S_iD}^2}}$=8.33% ($i$=1,2,4,5), and likewise p_{S_2}=8.33%, p_{S_4}=8.33%, and p_{S_5}=75.00%, respectively. Clearly, POW prefers to choose the shorter route with a higher probability. Here we have two individual cases to deal with: If the switch S_5 is chosen finally, then it will determinedly choose one of its neighbour nodes that is closest to the destination as the next-hop. Otherwise, if a normal node S_i, i=1,2,4, is selected as S's next-hop, then the above procedure is repeated taking S_i as the new source node until the packet reaches the destination.

B. Flow Control

The flow control mechanism is designed to manage the allocation of resources (e.g. queue buffer) to packets as they progress along their route so as to avoid congestion or deadlock. In the traditional Internet, protocols with flow control like TCP usually avoid congestion through dropping packets and retransmission. However, this mechanism is not suitable in the torus network due to its long routing path, e.g., if the

hot spot is very far from the source then dropping packet means the previous long transmission would have been of waste and the retransmission packets would occupy additional bandwidth which may make the network more congested. Thus, the most beneficial way is to achieve a packet lossless fabric, where the buffers of nodes in the network should be guaranteed no overflow. The common approach is to use credit-based flow control. As shown in Fig.6, the prerequisite for any uplink node sending data to a downlink node is that its output port has enough credits. The default value of credit is the number of packets that the downlink node can accept. The credits will be deducted by one whenever one packet is sent out. Correspondingly, whenever the downlink node vacates new buffer space to accept a new packet, it will send one credit to its uplink node and increase the credit of its corresponding port by one. Usually, there may exist a nearly negligible delay between packet transmission and credit feedback between two directly connected nodes, however, for safety the downlink node should have slightly larger buffer for several more packets to deal with any possible underflow issues. The credit can either be transferred via in-band or out-of-band signal, where the in-band way is more cost-effective but complicated to be implemented while out-of-band method is simple to be realized but with more cost. Besides, the Head-of-Line blocking issue can be addressed in the way of shared buffering where the output queue and input queue are logically organized as a shared virtual queue in the implementation.

C. Deadlock Avoidance Implementation

Deadlock occurs when a cycle of packets are waiting for one another to release resources (queue buffer) where the resource dependencies form a cycle. Packets are blocked indefinitely and throughput will also collapse at once. There are usually two deadlock avoiding techniques, including virtual channels (by decomposing each unidirectional physical channel into several logical channels with private buffer resources) and Turn Model Routing (by eliminating certain turns in some dimensions). To make our architecture deadlock free, we first ensure that the route cannot form a loop in POW routing. As aforementioned, each step of POW routing is guaranteed to be closer to the destination without jumping back, which ensures POW is deadlock free in the mesh sub-network. Finally, if packets go through the wraparound links in torus network, then as was done in [5] we use two virtual channels to cut the loop into two different logical channels, which is easy to be implemented and effectively avoids deadlock.

D. Livelock Prevention

Livelock is another notorious issue where packets continue to proceed through the network but do not advance towards their destination even though they are not blocked. This may occur when non-minimal adaptive routing is allowed where packets may be misrouted but are not able to get closer to their destination. In POW routing, it guarantees the packet delivery to destination in a greedy way, where the decision made at each step is based on the distance which requires $\Delta_{S_iD}<\Delta_{SD}$. Thus, POW routing enables packets to choose its next hop which is always closer to the destination than that from the current node, and with a high probability of choosing the shortest path. Therefore, we can claim that POW is a livelock-free routing algorithm.

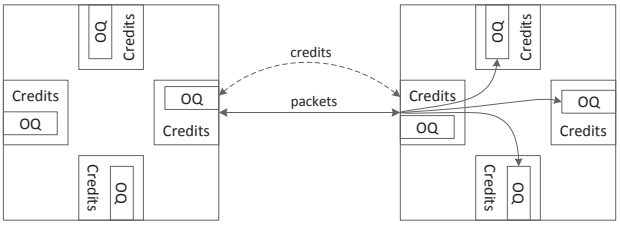

Fig. 6. Credit-based Flow Control implementation.

VI. Conclusion and Future Work

Based on torus topology, this paper presents a novel DCN architecture *CLOT* using a certain number of low-end switches to connect the most distant nodes in each dimension. By doing this, compared to a regular k-ary n-D torus network *CLOT* halves the network diameter, improves bisection bandwidth by at least 25% where the ratio increases with k, increases the ideal throughput to be at least 65%, and provides a much better path diversity resulting in a better fault tolerance. Additionally, coupled with the geographical coordinate address, *CLOT* can carry out the content routing which provides the possibility of implementing key-value stores in addition to the traditional routings. *CLOT* is also compatible with legacy TCP/IP protocols. Furthermore, under the specific flow control mechanism the probabilistic weighted routing algorithm POW helps *CLOT* achieve a better load balancing than the deterministic routing algorithms. Besides, POW is both a livelock and deadlock free algorithm. Due to the page space limitation, we only provide the theoretical analysis of *CLOT*'s performance with upper and lower boundaries, without any experiments. The evaluation of *CLOT* in real data center environment will be our future work.

VII. Acknowledgement

This paper was made possible in part supported by Huawei Technologies Co., Ltd..

References

[1] http://www.amazon.com/.
[2] M. A., et al. A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*. ACM, 2008.
[3] Hussam Abu-Libdeh, et al. Symbiotic routing in future data centers. *ACM SIGCOMM Computer Communication Review*, 41(4):51–62, 2011.
[4] Kai Chen, et al. Osa: An optical switching architecture for data center networks with unprecedented flexibility. 2012.
[5] W. J. Dally, et al. Deadlock-free adaptive routing in multicomputer networks using virtual channels. *TPDS*, 1993.
[6] W.J. Dally and B. Towles. *Principles and practices of interconnection networks*. Morgan Kaufmann, 2004.
[7] C. Guo, et al. DCell: A scalable and fault-tolerant network structure for data centers. *ACM SIGCOMM*, 38(4):75–86, 2008.
[8] Chuanxiong Guo, et al. Bcube: a high performance, server-centric network architecture for modular data centers. *ACM SIGCOMM*, 2009.
[9] Farrington Nathan, et al. Data Center Switch Architecture in the Age of Merchant Silicon. In *IEEE Hot Interconnects*, New York, Aug 2009.
[10] Ji-Yong Shin, et al. Small-world datacenters. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 2. ACM, 2011.
[11] Leslie G Valiant, et al. Universal schemes for parallel communication. In *the 13th annual ACM symposium on Theory of computing*, 1981.
[12] Ting Wang, et al. NovaCube: A Low Latency Torus-Based Network Architecture for Data Centers. IEEE GlobeCom, 2014.
[13] Ting Wang, et al. Sprintnet: A high performance server-centric network architecture for data centers. In *ICC*, pages 4005–4010. IEEE, 2014.