NovaCube: A Low Latency Torus-Based Network Architecture for Data Centers

Ting Wang, Zhiyang Su, Yu Xia, Bo Qin, Mounir Hamdi
Department of Computer Science and Engineering
Hong Kong University of Science and Technology, Hong Kong
{twangah, zsuab, rainsia, bqin, hamdi}@cse.ust.hk

***Abstract*—This paper presents the design, analysis, and implementation of a novel data center network architecture, named *NovaCube*. Based on regular Torus topology, *NovaCube* is constructed by adding a number of most beneficial jump-over links, which offers many distinct advantages and practical benefits. Moreover, in order to enable *NovaCube* to achieve its maximum theoretical performance, a probabilistic oblivious routing algorithm PORA is carefully designed. PORA is a both deadlock and livelock free routing algorithm, which achieves near-optimal performance in terms of average routing path length with better load balancing thus leading to higher throughput. Theoretical derivation and mathematical analysis further prove the good performance of *NovaCube* and PORA.**

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

I. Introduction

The data center network (DCN) architecture is regarded as one of the most important determinants of network performance in data centers, and plays a significant role in meeting the requirements of could-based services as well as the agility and dynamic reconfigurability of the infrastructure for changing application demands. As a result, many novel proposals, such as Fat-Tree [1], DCell [3], BCube [12], c-Through [11], Helios [14], SprintNet [20], FlatNet [5], Hyper-BCube [6], CamCube [2], Small-World [13], and so on, have been proposed aiming to efficiently interconnect the servers inside a data center to deliver peak performance to users.

Generally, DCN topologies can be classified into four categories: tree-based topology (e.g. Fat-Tree), recursive-defined topology (e.g. DCell, BCube, SprintNet), hybrid network (e.g. c-Through, Helios) and direct network (e.g. CamCube, Small-World). Each of these has their advantages and disadvantages. Tree-based topologies, like FatTree and Clos, can provide full bisection bandwidth, thus the any-to-any performance is good. However, its scalability is limited and its building cost and complexity is relatively high. The recursive-defined topology usually concentrates on the scalability and incremental extensibility with a lower building cost; however, the full bisection bandwidth may not be achieved and their performance guarantee is only limited to a small scope. The hybrid network is a hybrid packet and circuit switched network architecture. Compared with packet switching, optical circuit switching can provide higher bandwidth and lower latency in transmission with lower energy consumption. However, optical circuit switching cannot achieve full bisection bandwidth at packet granularity. Furthermore, the optics also suffers from slow switching speed which can take as high as tens of milliseconds.

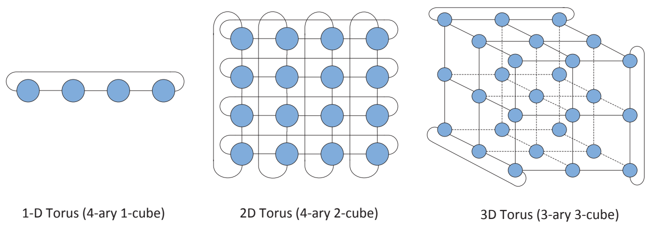

Fig. 1. Examples of 1D, 2D, 3D Torus topologies.

The direct network topology, which directly connects servers to other servers, is a switchless network interconnection without any switches, or routers. It is usually constructed in a regular pattern, such as Torus (as show in Fig.1). Besides being widely used in high-performance computing systems, Torus is also an attractive network architecture candidate for data centers. However, this design suffers consistently from poor routing efficiency compared with other designs due to its relatively long network diameter (the maximum shortest path between any node pairs), which is known as $\left\lfloor \frac{k}{2} \right\rfloor n$ for a n-D Torus with radix k. Besides, a long network diameter may also lead to high communication delay. Furthermore, its performance largely depends on the routing algorithms.

In order to deal with these imperfections, in this paper we propose a novel container level high-performance Torus-based DCN architecture named *NovaCube*, which can halve the network diameter and receive higher bisection bandwidth and throughput. Moreover, we design a new weighted probabilistic oblivious deadlock-free routing algorithm PORA for *NovaCube*, which achieves low average routing path length and good load-balancing by exploiting the path diversities.

The rest of the paper is organized as follows. First we briefly review the related research literature in Section II. Then Section III demonstrates the motivation. In Section IV, *NovaCube* architecture is introduced and analyzed in detail. Afterwards, the routing algorithm PORA is designed in Section V. Finally, Section VI concludes this paper.

II. Related Work

The Torus-based topology well implements the network *locality* forming the servers in close proximity of each other, which increases the communication efficiency. Besides being widely used in supercomputing, torus network has also been introduced to the data center networks. Two typical representatives are namely CamCube [2] and Small-World [13].

CamCube is proposed by Abu-Libdeh et al., and the servers in CamCube are interconnected in a 3D torus topology. The CamCube is designed target to shipping container-sized data centers, and is a server-only switchless network design. With the benefit of Torus architecture and the flexibility offered by a low-level link orientated CamCube API, CamCube allows applications to implement their own routing protocols so as to achieve better application-level performance. However, as aforementioned this design based on the regular Torus suffers long average routing path – $O(N^{1/3})$ hops, with N servers, which results in poor routing efficiency.

In order to overcome this limitation, Shin Ji-Yong, et al. proposed Small-World, which provides an unorthodox random data center network topology. It is constructed based on some regular topologies (such as ring, torus or cube) with the addition of a large number of random links which can help reduce the network diameter and achieve higher routing efficiency. The degree of each node in Small-World is limited to six, taking realistic deployment and low cost into consideration. In addition to traditional routing methods, Small-World also provides content routing coupled with geographical address assignment, which in turn efficiently implements key-value stores. However, its shortest path routing suffers poor worst-case throughput and poor load balancing.

III. Motivation

A. Why Torus-based Clusters

The Torus (or precisely k-ary n-cube) based interconnection has been regarded as an attractive DCN architecture scheme for data centers because of its own unique advantages, some of which are listed below.

Firstly, it incurs lower infrastructure cost since it is a switchless architecture without needing any expensive switches. In addition, the power consumed by the switches and its associated cooling power can also be saved.

Secondly, it achieves better fault-tolerance. Traditional architecture is usually constructed with a large number of switches, any failure of which could greatly impact on the network performance and system reliability. For example, if a ToR switch fails, the whole rack of servers will lose connection with the servers in other racks. Comparatively, the rich interconnectivity and in-degree of Torus-based switchless architecture makes the network far less likely to be partitioned. The path diversity can also provide good load balance even on permutation traffic.

Thirdly, the architectural symmetry of Torus topology optimizes the scalability and granularity of Clusters. It allows systems to economically scale to tens of thousands of servers, which is well beyond the capacity of Fat-Tree switches. For an n-ary k-cube, the network can support up to k^n nodes, and scales at a high exponential speed which outperforms traditional switched networks, such as Fat-Tree's $O(p^3)$, and BCube's $O(p^2)$, where p denotes p-port switch.

Fourthly, Torus is also highlighted by its low cross-cluster latency. In traditional switched DCNs, an inevitably severe problem is that the switching latency (several μs in each hop) and the TCP/IP processing latency (tens of μs) are very high, which leads to a long RTT. Comparatively, TCP/IP stack is not needed in Torus network which saves the long TCP/IP processing time, and the NIC processing delay is also lower than switches (e.g., the processing delay of a real VirtualShare NIC engine is only 0.45 μs). Besides, Torus also avoids the network oversubscription and provides many equal cost routing paths to avoid network congestion, which can help reduce the queuing delay due to network congestion. Consequently, Torus achieves a much lower end-to-end delay, which is very important in the data center environment.

Fifthly, its high network performance has already been proven in high-performance systems and supercomputers, such as Cray Gemini (3D Torus) [17], IBM's Blue Gene/L (3D Torus) [16] and Blue Gene/Q (5D Torus) [4].

B. Routing Issues in Torus

Any well qualified routing algorithm design in Torus network must take all important metrics (such as throughput, latency, average path length, load balancing, deadlock free) into consideration. However, the current existing routing algorithms in Torus are far from perfect, as when they improve some certain performance it is usually at the sacrifice of others. Generally the routings in Torus can be divided into two classes: deterministic routing and adaptive routing. A common example of deterministic routing is dimension-ordered routing (DOR) [7], where the message routes dimension by dimension and the routing is directly determined by the source address and destination address without considering the network state. DOR achieves a minimal routing path, but also eliminates any path diversity provided by torus topology, which results in poor load balancing and low throughput. As an improved two-phrase DOR algorithm, Valiant routing (VAL) [19] can achieve optimal worst-case throughput by adding a random intermediate node, but it destroys locality and suffers longer routing path. ROMM [15] and RLB [18] implements good locality, but cannot achieve optimal worst-case throughput. Comparatively, the adaptive routing (like MIN AD [10]) uses local network state information to make routing decisions, which achieves better load balancing and can be coupled with a flow control mechanism. However, using local information can lead to non-optimal choices while global information is more costly to obtain, and the network state may change rapidly. Besides, adaptive routing is not deadlock free, where a resource cycle can occur without routing restrictions which leads to a deadlock. Thus, adaptive routings have to apply some dedicated deadlock-avoiding techniques, such as Turn Model Routing (by eliminating certain turns in some dimensions) and Virtual Channels (by decomposing each unidirectional physical link into several logical channels with private buffer resources), to prevent deadlock.

To summarize, the good features of Torus conclusively demonstrate its superiority in constructing a cost-effective and high performance data center network. However, it also suffers some shortcomings, such as the relatively long routing path, and inefficient routing algorithm. In response to these issues, in this paper we propose some practical and efficient solutions from the perspectives of physical interconnection and routing.

IV. NovaCube Network Design

This section presents the network design and theoretical analysis of *NovaCube*. Before introducing the physical interconnection structure, we firstly provide a theorem with proof, which offers a theoretical basis of *NovaCube* design.

Theorem 3.1. *For any node A(a_1, a_2, ..., a_n) in a k-ary n-cube (when k is even) if B(b_1, b_2, ..., b_n) is assumed to be the*

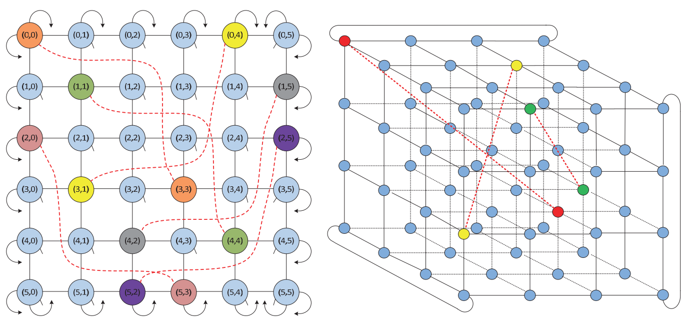

Fig. 2. A 2D 6×6-node and 3D 4×4×4-node *NovaCube* (for simplicity, not all jump-over links are shown).

farthest node from A, then B is unique and B's unique farthest node is exactly A.

Proof. In a k-ary n-cube, 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 there is:

$$b_1 = (a_1 + \frac{k}{2})\ mod\ k,\ \ldots,b_n = (a_n + \frac{k}{2})\ mod\ k \qquad (1)$$

Since the result of $(a_i + \frac{k}{2})\ mod\ k$ is unique, thus $\forall b_i$ is unique and $b_i \in [0, k)$. Hence, A's farthest node B is unique.

Next, assume B's farthest node is A'(a'_1, a'_2, ..., a'_n), similarly we have:

$$a'_1 = (b_1 + \frac{k}{2})\ mod\ k,\ \ldots,a'_n = (b_n + \frac{k}{2})\ mod\ k \qquad (2)$$

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

$$a'_i = (b_i + \frac{k}{2})\ mod\ k = \{[(a_i + \frac{k}{2})\ mod\ k] + \frac{k}{2}\}\ mod\ k$$
$$\because a_i \in [0, k), \quad \therefore a_i + \frac{k}{2} \in [\frac{k}{2}, k + \frac{k}{2})$$

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

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

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

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

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. □

A. NovaCube Physical Structure

As aforementioned, one critical drawback of k-ary n-cube topology is its relatively long network diameter, which is as high as $\lfloor \frac{k}{2} \rfloor n$. In order to decrease the network diameter and make routing packets to far away destinations more efficiently, based on the regular k-ary n-cube, *NovaCube* is constructed by adding some jump-over links connecting the farthest node pairs throughout the network. In a n-D Torus the most distant node of (a_1, a_2, ..., a_n) can be computed as $((a_1 + \lfloor \frac{k}{2} \rfloor)\ mod\ k, (a_2 + \lfloor \frac{k}{2} \rfloor)\ mod\ k, ..., (a_n + \lfloor \frac{k}{2} \rfloor)\ mod\ k)$, which guides the construction of *NovaCube*. In brief, the key principle of *NovaCube* is to connect the most distant node pairs by adding one jump-over link. More precisely, there are two construction cases with tiny differences.

1) Case 1: when k is even, then according to Theorem 3.1 any node's farthest node is unique to each other, and there are $\frac{k^n}{2}$ farthest node pairs, where k^n is the total number of nodes. In this case, all the $\frac{k^n}{2}$ farthest node pairs are connected to each other by one jump-over link. As a result, the degree of each node will be increased from original $2n$ to $2n + 1$. Fig.2 presents two examples of 2D and 3D *NovaCube*.

2) Case 2: when k is odd, one node's farthest node cannot be guaranteed to be unique, nor is the number of node pairs $\frac{k^n}{2}$ an integer either since k^n is odd. In consideration of this fact, we have no alternative but to settle for the second-best choice. The eclectic way is to only construct (k-1)-ary n-*NovaCube*, and keep the k-th node in each dimension unchanged. Noticing that k ($k \geq 1$) is odd, then $k - 1$ is even. Therefore, the construction of n-D *NovaCube* with radix $k - 1$ is the same as Case 1. Consequently, there are $\frac{(k-1)^n}{2}$ node pairs with node degree of $2n + 1$ that are connected, and $n^k - n^{k-1}$ nodes with node degree of $2n$ remain unchanged. This way makes a trade-off, however a small one.

B. Properties of NovaCube

As with any network, the performance of the *NovaCube* network is characterized by its network diameter, bisection bandwidth, throughput, path diversity and physical cost.

1) Network Diameter

After connecting the most distant node pairs by additional jump-over links, the *NovaCube* network architecture halves the diameter, where the diameter is reduced from original D_{Torus} = $\lfloor \frac{k}{2} \rfloor n$ to be current

$$D_{NovaCube} = \left\lceil \frac{\lfloor \frac{k}{2} \rfloor n}{2} \right\rceil \qquad (3)$$

Proof. The network diameter of a regular k-ary n-cube (n-D Torus) is D_{Torus} = $\lfloor \frac{k}{2} \rfloor n$, which means that any node inside the network can reach all the other nodes within $\lfloor \frac{k}{2} \rfloor n$ hops. For any node A in Torus, we assume that node B is the farthest node from node A. Next, we assume set S_i to denote the nodes that can be reached from node A at the i-th hop in Torus, where $i \in [0, \lfloor \frac{k}{2} \rfloor n]$. Then the universal set of all nodes in the network can be expressed as $S = \sum_{i=0}^{\lfloor \frac{k}{2} \rfloor n} S_i$.

After linking all the most distant node pairs (e.g. A and B) in *NovaCube*, if we define S'_i as the set of nodes that are i hops from node A in *NovaCube*, then: (1) for the case of $\lfloor \frac{k}{2} \rfloor n$ is even, we have $S'_0 = S_0$, $S'_1 = S_1 + S_{\lfloor \frac{k}{2} \rfloor n}$, $S'_2 = S_2 + S_{\lfloor \frac{k}{2} \rfloor n-1}$, $\ldots$,$S'_{\lfloor \frac{k}{2} \rfloor n/2} = S_{\lfloor \frac{k}{2} \rfloor n/2} + S_{\lfloor \frac{k}{2} \rfloor n/2+1}$; (2) for the case of $\lfloor \frac{k}{2} \rfloor n$ is odd, we have $S'_0 = S_0$, $S'_1 = S_1 + S_{\lfloor \frac{k}{2} \rfloor n}$, $S'_2 = S_2 + S_{\lfloor \frac{k}{2} \rfloor n-1}$, $\ldots$,$S'_{\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil -1} = S_{\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil -1} + S_{\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil +1}$, $S'_{\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil} = S_{\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil}$. This demonstrates that in *NovaCube* any node A can reach all nodes of the entire network within $\lfloor \frac{k}{2} \rfloor n/2$ or $\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil$ hops. Consequently, the network diameter of *NovaCube* is $\lceil \lfloor \frac{k}{2} \rfloor n/2 \rceil$. This ends the proof. □

2) Bisection Bandwidth

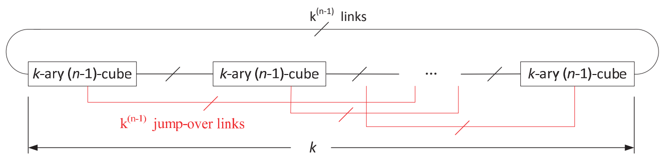

Fig. 3. A k-ary n-*NovaCube*.

The bisection bandwidth can be calculated by summing up the link capacities between two equally-sized parts which the network is partitioned into. It can be used to evaluate the worst-case network capacity [9]. Assume the *NovaCube* network $T(N_1,N_2)$ is partitioned into two equal disjoint sets N_1 and N_2, each element of $T(N_1,N_2)$ is a bidirectional channel with a node in N_1 and another node in N_2. Then the number of bidirectional channels in the partition is $|T(N_1,N_2)|$, or $2|T(N_1,N_2)|$ channels in total, thus the bisection bandwidth is $B_T = 2 \mid T(N_1, N_2) \mid$. For a k-ary n-*NovaCube* as shown in Fig.3, when k is even, there is even number of k k-ary (n-1)-cube, which can be divided by the minimum bisection into two equal sets with $2k^{n-1}$ regular bidirectional links and $k * \frac{k^{n-1}}{2}$ jump-over bidirectional links. Therefore, the channel bisection bandwidth of k-ary n-*NovaCube* is computed as:

$$B_T = 2*(2k^{n-1}+k*\frac{k^{n-1}}{2}) = k^n+4k^{n-1} \tag{4}$$

According to the result in [8], the bisection bandwidth of a regular n-dimensional Torus with radix k is $B_C = 4k^{n-1}$. Therefore, *NovaCube* effectively increases the bisection bandwidth by at least $\frac{B_T - B_C}{B_C} = \frac{k^n+4k^{n-1}-4k^{n-1}}{4k^{n-1}} = \frac{k}{4} \geq 25\%$ (k≥1) and the ratio increases accordingly as k increases.

3) Throughput

Throughput is a key indicator of the network capacity to measure a topology. It not only largely depends on the bisection bandwidth, but is also determined by the routing algorithm and flow control mechanism. However, we can evaluate the ideal throughput of a topology under the assumed perfect routing and flow control. The maximum throughput means some channel in the network is saturated and the network cannot carry more traffic. Thus, the throughput is closely related to the channel load. We assume the bandwidth of each channel is b_c and the workload on a channel c is ω_c. Then the maximum channel load ω_{max} = Max$\{\omega_c, c \in C\}$. The ideal throughput occurs when the bottleneck channel is saturated and equal to the channel bandwidth b_c. Thus, the ideal throughput Θ_{ideal} of a topology is

$$\Theta_{ideal} = \frac{b_c}{\omega_{max}} \tag{5}$$

Under uniform traffic pattern, the maximum channel load ω_{max} at the bisection channel has a lower bound, which in turn gives an upper bound on throughput. For a uniform traffic pattern, on average $\frac{k^n}{2}$ packets must go through the B_T bisection channels. If the routing and flow control are optimal, then the packets will be distributed evenly among all bisection channels which results in the best throughput. Thus, the load on each bisection channel load is at least

$$\omega_{max} \geq \frac{\frac{k^n}{2}}{B_T} = \frac{\frac{k^n}{2}}{k^n+4k^{n-1}} = \frac{k}{2k+8} \tag{6}$$

Consequently, the upper bound on an ideal throughput under uniform traffic can be derived from Equations (5) and (6):

$$\Theta_{ideal} = \frac{b_c}{\omega_{max}} \leq \frac{2k+8}{k} b_c \tag{7}$$

This exhibits that *NovaCube* achieves better performance than regular Torus topology in the network capacity with respect to throughput, where the ideal throughput of Torus is only $8b_c/k$ [8]. Here we normalize the worst-case throughput $\widehat{\Theta}_{nw}$ to the network capacity: $\widehat{\Theta}_{nw} = \frac{\omega_{max}}{\omega_{nw}(\widehat{R})}$, where $\widehat{R}$ indicate a routing algorithm. Valiant routing (VAL) [19] is a known worst-case throughput optimal routing algorithm in Torus which obtains $\widehat{\Theta}_{nw}$=50%. Thus, an optimal routing algorithm in *NovaCube* can achieve normalized worst-case throughput of at least $\widehat{\Theta}_{nw}$=62.5%.

4) Path Diversity

Inherited from Torus topology, *NovaCube* provides a diversity of paths, which can be exploited in routing algorithm by selectively distributing traffic over these paths to achieve load balancing. Besides, the network reliability also greatly benefits much from the path diversity, where the traffic can route around the faulty nodes/links by taking alternative paths.

The number of distinct paths existing in *NovaCube* is too huge to be calculated exactly, for simplicity, we first compute the number of shortest paths in a regular Torus without any jump-over links. Assume two nodes A$(a_1, a_2, \ldots, a_n)$ and B$(b_1, b_2, \ldots, b_n)$ in an n-dimensional Torus, and the coordinate distance between A and B in the i^{th} dimension is Δ_i = $|a_i - b_i|$. Then the total number of shortest paths P_{ab} between A and B is:

$$P_{ab} = \prod_{i=1}^{n} \binom{\sum_{j=i}^{n} \Delta_j}{\Delta_i} = \frac{(\sum_{i=1}^{n} \Delta_i)!}{\prod_{i=1}^{n} \Delta_i!} \tag{8}$$

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 in dimension i out of all the remaining hops. It can be seen that a longer distance and a higher dimension result in a larger number P_{ab} of shortest paths. For instance, if given Δ_x = 3, Δ_y = 4, Δ_z = 5 in a 3D Torus, the number of shortest paths is as high as 27720. If we further add a larger number of additional jump-over links in *NovaCube*, the number of paths will be larger. If taking the non-minimal paths into consideration as designed in some routing algorithms, the number of feasible paths is nearly unlimited. The great path diversity of *NovaCube* offers many benefits as aforementioned, but it is also confronted with great challenges in designing an efficient, deadlock free and load balanced routing algorithm.

5) Average Path Length

In this subsection, we derive the average path length (APL) for n-D *NovaCube* with an even radix k, and calculate its value for n=2. Due to the symmetry, every node is the same in the k-ary n-*NovaCube*. Hence, we can only consider the APL from a fixed source s to any possible destination d. Here we denote m as the jump-over neighbour of s.

Denote source s=$(0, ..., 0)$ and destination d=$(x_1, ..., x_n)$, where $x_i \in [0, k-1]$. Then we have m=$(\frac{k}{2}, ..., \frac{k}{2})$. Thus, the

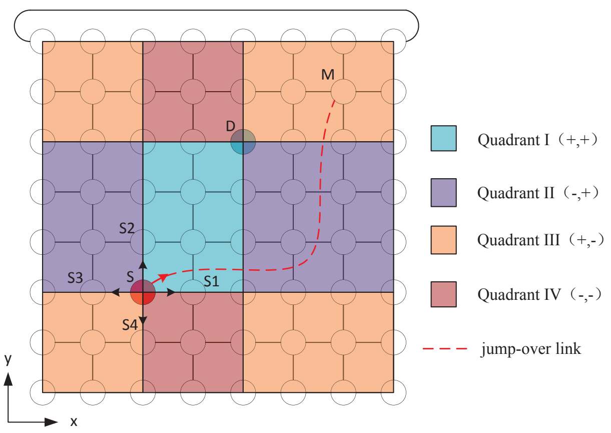

Fig. 4. PORA in an 8×8 *NovaCube* (for simplicity, not all wraparound links and jump-over links are displayed).

s-d minimal distance in k-ary n-*NovaCube* is given as:

$$\Delta(s,d) \overset{\text{def}}{=} \min\{||s-d||_1, ||m-d||_1+1\} \tag{9}$$

where $||\cdot||_p$ is p-norm of the vector meaning that $||x||_p = (\sum_{i=1}^{n}|x_i|^p)^{\frac{1}{p}}$ for the n-dimensional vector x. Then, the APL is $E[\Delta(s,d)]$. Without loss of generality, for the case $n=2$, we can have

$$\begin{aligned} E[\Delta(s,d)] &= \frac{1*5+\sum_{i=2}^{k/2-1}(8i-4)*i+(4k-6)*\frac{k}{2}}{k^2-1} \\ &= \frac{\frac{k^3}{3}+\frac{k^2}{2}-\frac{4k}{3}+1}{k^2-1} \end{aligned} \tag{10}$$

Thus, the APL of *NovaCube* approaches to $\frac{k}{3}$ when k is large, which is superior to 2D Torus's $\frac{k}{2}$ [8], and as the dimension increase *NovaCube* reduces more. In the similar way, we can compute the APL for the k-ary n-D *NovaCube*.

V. Routing Scheme

This section presents the specially designed routing algorithms named PORA for *NovaCube*, which aims to help *NovaCube* achieve its maximum theoretical performance. PORA is a probabilistic weighted oblivious routing algorithm. Besides, PORA is also livelock and deadlock free.

A. PORA Routing Algorithm

Notation 1. The distance between node A$(a_1, a_2, \ldots, a_n)$ and node B$(b_1, b_2, \ldots, b_n)$ in the i-th dimension is denoted as $\Delta_i = ||a_i - b_i||_1$. The distance between A and B is given as $\Delta_{AB} = \sum_{i=1}^{n}\Delta_i$.

Generally, the PORA procedure can be divided into two steps. The first step is to choose routing direction according to the given probability, while the second step is to route within the designated quadrant. Without loss of generality, for simplicity we use 2D *NovaCube* to illustrate PORA.

1) Step 1: As shown in Fig.4, assume a packet needs to route from the source node S to the destination node D, then firstly it needs to decide the direction of its first hop. Since S has five neighbour nodes S_1, S_2, S_3, S_4, M, where M is its jump-over neighbour (although in the case of odd k some special nodes may have no jump-over links, PORA still works correctly), thus it has five directions to route the packet. In order to choose the most beneficial next-hop, each direction is assigned a probability based on the distance between S's next-hop node and destination node. Then PORA chooses the next-hop according to their probabilities, where the probabilistic mechanism can help PORA achieve a good load balancing. The normalized probability function is given as below:

$$p_i = \frac{\frac{1}{\Delta_i^2}}{\sum_{i=1}^{\psi}\frac{1}{\Delta_i^2}} \tag{12}$$

where ψ is the number of neighbour nodes of the source. Take Fig.4 as an example, the distances between S's neighbour nodes and destination node D are Δ_{S_1D}=4, Δ_{S_2D}=4, Δ_{S_3D}=6, Δ_{S_4D}=6, Δ_{MD}=3, thus the probability of choosing S_1 as the next-hop is $p_{S_1}=\frac{\frac{1}{\Delta_{S_1D}^2}}{\sum\frac{1}{\Delta_i^2}}$=21.43%, and likewise p_{S_2}=21.43%, p_{S_3}=9.524%, p_{S_4}=9.524%, p_M=38.10%, respectively. Clearly, PORA prefers to choose the shorter path with a higher probability. Each neighbour node (except the jump-over neighbour) corresponds to one quadrant in the Cartesian coordinate system as shown in Fig.4. For example, in Fig.4 S_1, S_2, S_3, S_4 correspond to Quadrant I, II, III, and IV, respectively. The division of quadrants is only determined by the source node and destination node.

2) Step 2: There are two cases for this step. For the first, if Step 1 finally selects the regular neighbour node other than the jump-over neighbour as the next hop, then all the following routing decisions towards the destination must be restricted within the corresponding quadrant. For the second, in case Step 1 chooses the jump-over neighbour node M (e.g. if it has a smaller distance thus with a higher probability to be chosen) as the next hop, then repeat Step 1 to determine the quadrant by taking M as the source node. This time, in Step 1 PORA will only compute the probability of its regular neighbours without considering its jump-over neighbour, which can avoid jumping back to the original source node that may result in livelock issue.

Once the quadrant is determined, then PORA routes the packet only within the chosen quadrant, where the routing mechanism applied is also probabilistic. At each hop, PORA firstly check if the jump-over link can be used. The jump-over hop can be taken as the candidate next-hop route if and only if it satisfies the following two requirements:

- The jump-over neighbour node is also located within the same quadrant.
- The distance between jump-over neighbour node and destination node is smaller than the distance between regular neighbour node and destination node.

If the requirements cannot be satisfied, then PORA will take the regular neighbour node as its next-hop using traditional DOR (Dimension-Ordered Routing [7]) algorithm, which routes the packet dimension by dimension. Otherwise, if the jump-over link meets the requirements, then the next-hop is selected from the jump-over node and DOR node according to the probability as computed in Equation (12). This process is repeated at each hop until it reaches the destination.

B. Livelock Prevention

Livelock occurs when a packet is denied routing to its destination forever even though it is never blocked permanently. It may be travelling around its destination node,

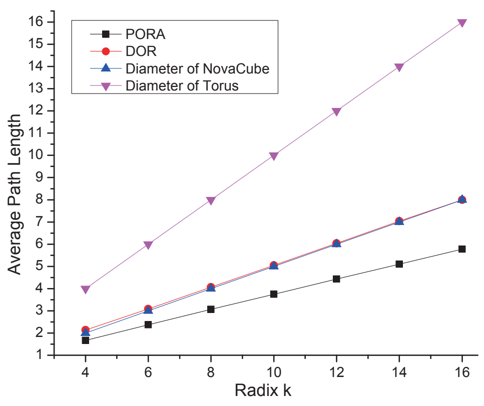

Fig. 5. The average path length using PORA and DOR.

never reaching it because the channels it requires are always occupied by other packets. This can occur if non-greedy adaptive routing is allowed (packets are misrouted, but are not able to get closer to the destination). In PORA, once the routing direction is determined at the first step, each of the following hops of PORA will be restricted within the selected quadrant. Moreover, the routing method within the quadrant enables the packet to find its next hop, whose distance to the destination node is always smaller than that from the current node, which guarantees packet delivery. Thus, we claim that PORA is a livelock-free routing algorithm.

C. Deadlock-Free Implementation

As an another notorious problem in Torus networks, deadlock is the situation where packets are allowed to hold some resources while requesting others, so that the dependency chain forms a cycle. Then all these packets must wait forever without reaching the destination, and the throughput will also collapse. The DOR algorithm is proven to be deadlock-free in a mesh network, since there will be no message loop in the network. However, the Torus message loops by itself, thus simply using DOR cannot prevent deadlock. Virtual channels are proposed as a very effective means to prevent deadlock from happening. Virtual channels are used in the loops in a network to cut the loop into two different logical channels, so no cyclic dependency will be formed. Virtual channel is easy to implement by using multiple logical queues and effective in solving deadlock.

To prevent deadlock in our architecture, we first make sure that jump-over links cannot form loops in the routing. We ensure that any jump-over links we choose in the quadrant must be nearer than the previous hop and regular torus links towards the destination; otherwise, regular torus links are used. This ensures that the packet will never jump back through the jump-over links. Then, we use the DOR routing to prevent the deadlock in the mesh sub-network. Finally, if the packets have to pass through the wraparound links in the Torus network, we use two virtual channels to cut a Torus loop into two different logical paths to avoid the deadlock. Thus, only two virtual channels are needed in each direction, which is still cost-effective, since the hardware cost increases as the number of virtual channels increases.

D. Average Path Length

Fig.5 exhibits the simulation results of average path length (APL) using PORA in *NovaCube* and DOR (known to be a shortest path routing algorithm) in Torus under uniform traffic distribution mode. The result reveals that PORA indeed achieves a smaller APL in *NovaCube* than DOR in regular Torus, where a smaller APL implies a lower network latency. Even the network diameter of *NovaCube* is also slightly smaller than the achieved APL by DOR in Torus. Moreover, the APL achieved by PORA is already very close to the theoretical analysis, which demonstrates the optimality of PORA.

VI. Conclusion

In this paper, we proposed a novel data center architecture named *NovaCube*, and presented its design and key properties. As a switchless architecture, *NovaCube*'s cost-effectiveness is highlighted with regard to its energy consumption and infrastructure cost. As proved *NovaCube* is also superior to other candidate architectures in terms of network diameter, throughput, average path length, bisection bandwidth, path diversity and fault tolerance. Furthermore, the specially designed probabilistic weighted oblivious routing algorithm PORA helps *NovaCube* achieve near-optimal average path length with better load balancing which can result in a better throughput. Moreover, PORA is also free of livelock and deadlock.

VII. Acknowledgement

This paper is supported in part by HKUST RGC 612912, and Huawei Technologies Co., Ltd.

References

[1] M. A., et al. A scalable, commodity data center network architecture. In *ACM SIGCOMM Computer Communication Review*. ACM, 2008.
[2] H. Abu-Libdeh, et al. Symbiotic routing in future data centers. *ACM SIGCOMM Computer Communication Review*, 40(4):51–62, 2010.
[3] Guo C., et al. DCell: A scalable and fault-tolerant network structure for data centers. *ACM SIGCOMM*, 38(4):75–86, 2008.
[4] Chen. D., et al. The ibm blue gene/q interconnection fabric. *IEEE Micro*, 32(1):0032–43, 2012.
[5] Lin D, et al. FlatNet: Towards A Flatter Data Center Network. IEEE GLOBECOM, 2012.
[6] Lin D., et al. HyperBCube: A Scalable Data Center Network. IEEE ICC 2012, 2012.
[7] W. J. Dally, et al. Deadlock-free adaptive routing in multicomputer networks using virtual channels. *TPDS*, 1993.
[8] W.J. Dally and B. Towles. *Principles and practices of interconnection networks*. Morgan Kaufmann, 2004.
[9] Nathan Farrington, et al. Data Center Switch Architecture in the Age of Merchant Silicon. In *IEEE Hot Interconnects*, New York, Aug 2009.
[10] Luis G., et al. Adaptive deadlock-and livelock-free routing with all minimal paths in torus networks. *IEEE TPDS*, 5(12):1233–1251, 1994.
[11] Wang G., et al. c-Through: Part-time optics in data centers. In *SIGCOMM*, volume 40, pages 327–338. ACM, 2010.
[12] Chuanxiong Guo, et al. Bcube: a high performance, server-centric network architecture for modular data centers. *ACM SIGCOMM*, 2009.
[13] Shin Ji-Yong, et al. Small-world datacenters. In *Proceedings of the 2nd ACM Symposium on Cloud Computing*, page 2. ACM, 2011.
[14] Farrington N., et al. Helios: a hybrid electrical/optical switch architecture for modular data centers. In *ACM SIGCOMM*, 2010.
[15] Ted Nesson, et al. Romm routing on mesh and torus networks. In *Proceedings of the 7th annual ACM SPAA*, pages 275–287, 1995.
[16] Adiga N.R., et al. Blue gene/l torus interconnection network. *IBM Journal of Research and Development*, 49(2.3):265–276, 2005.
[17] A. Robert, et al. The gemini system interconnect. In *HOTI*. IEEE, 2010.
[18] Arjun Singh, et al. Locality-preserving randomized oblivious routing on torus networks. In *the 14th annual ACM SPAA*, 2002.
[19] Leslie G Valiant, et al. Universal schemes for parallel communication. In *the 13th annual ACM symposium on Theory of computing*, 1981.
[20] Ting Wang, et al. SprintNet: A High Performance Server-Centric Network Architecture for Data Centers. IEEE ICC, 2014. to appear.