On the design and analysis of Wireless Deadline-aware SWCube for data center networking

Senhong Huang and Xiaohua Tian
Shanghai Jiao Tong University, Shanghai, China
{huangsenhong,xtian}@sjtu.edu.cn

Abstract—We consider the design and analysis of Wireless Deadline-aware SWCube for data center networking to interconnect dual-port servers and commodity switches.The 60 GHz wireless technology is introduced to our architecture to relieve hotspots and to improve performance. Further,we explore a novel congestion avoidance algorithm, Wireless Deadline-aware TCP, to meet the requirements for data center networks: low latency for short flow, high burst tolerance and high utilization for long flows.

Index Terms—60GHz technology, SWCube, wireless deadline-aware TCP, data center.

1 INTRODUCTION

DATA centers are becoming increasingly important infrastructures to provide online applications services, such as search, e-mails,IMs, web2.0, and gaming, ect. In addition, these data centers also host infrastructure services such as GFS [1], HDFS [2], Bigtable [3], MapReduce [4] and Dryad [5]. As the increasing of the service demand will never end, the number of servers in today's data centers is required to be very large, for example, hundreds of thousands or millions.

While those diverse applications and mixing workloads generate a mix of short and long flows in data centers and require low latency for short flows,high burst tolerance and high utilization for long flows [6].Traditional TCP protocol falls short. We then consider two fundamental problem: how to design network architectures to connect large numbers of servers? And how to design a novel transport protocol to meet the needs of applications in data centers?

In this paper, we propose Wireless Deadline-aware SWCube to address those impairments. The design and implementation of Wireless SWCube are driven by existing works: Hypercube network [7], 60 GHz wireless technology [8], and the DCTCP protocol [6]. First, we observe that the commodity servers used in today's data centers usually come with two built-in Ethernet ports. Can we build a low-cost network using 2-NIC-port servers and low-end, multi-port commodity switches? A scalable solution is replace the nodes in the original generalized hypercube with switches and insert one server into each link [9]. Second, the burst of query traffic in data centers leads to hotspots. However, only few link pairs are hot and it is unpredictable [8]. Therefore, we explore the using of 60 GHz wireless technology and modification of state-of-art TCP protocol to satisfy the latency requirement and throughput performance of flows in data center. In summary,we make two main contribution in this work: designing a novel structure, Wireless SWCube and proposing deadline-aware TCP algorithm for this topology.

The rest of the paper is organized as follows. In Section 2, we discuss some related work. we propose our design, the Wireless SWCube for data center in Section 3. In Section 4,We then introduce our Deadline-aware TCP algorithm for the wireless network. Conclusions and future work are sketched in Section 5.

2 RELATED WORK

2.1 Data Center Network Topologies

Typical architectures(see Fig. 1) today consist of three-layer trees of switches or routers, core and the access routers tier in the root of the tree,an aggregation and access switches tier in the middle and top of rack switches tier at the leaves.Unfortunately, for the bottleneck of core routers in the layer 3, this conventional design suffers from four fundamental limitations: bandwidth limited, low scalability, low reliability and high cost.

To meet the requirements of high bandwidth, high scalability, burst tolerance and low cost, a variety of topologies are presented. We have made a survey of these architectures presented from 2008 to 2014 and proposed classification. Considering physical interconnection characteristics and structure construction, these architectures fall into four categories, as shown in the Fig. 2. The first category is tree-based topology, namely Fat-Tree [10], PortLand [11], VL2 [12], ElasticTree [13], AspenTree [14]; the second is recursively hierarchical topology, such as DCell [15], FiConn [16], BCube [17],

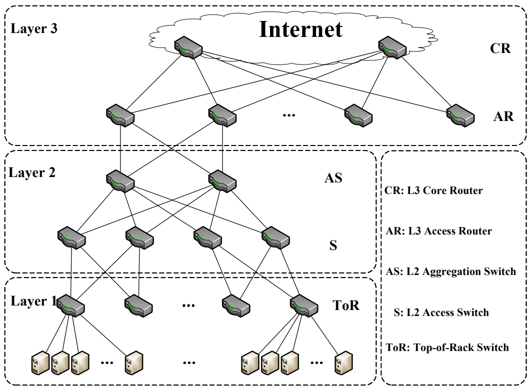

Fig. 1: Conventional architectures for data centers.

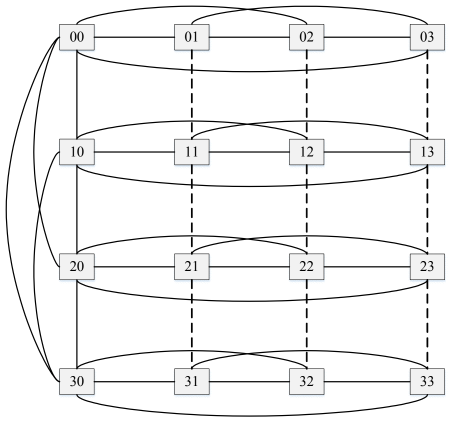

Fig. 3: The 2D generalized Hypercube architecture.

PCube [18], HCN [19], BCN [19] and Snowflake [20]; the third category is wireless topology, that is, Cayley Completely Wireless topology [21], WDCN [22] and 3D Beamforming Hybrid DCN [23]; the last category is other kind, such as Monsoon [24], DPillar [25] and Jellyfish [26]. There is no denying the fact that all these structures mentioned above outperform traditional three-tier architecture. However, our goal is to design data center network for interconnecting dual port servers and commodity switches, and feasible for the employment of wireless technology.

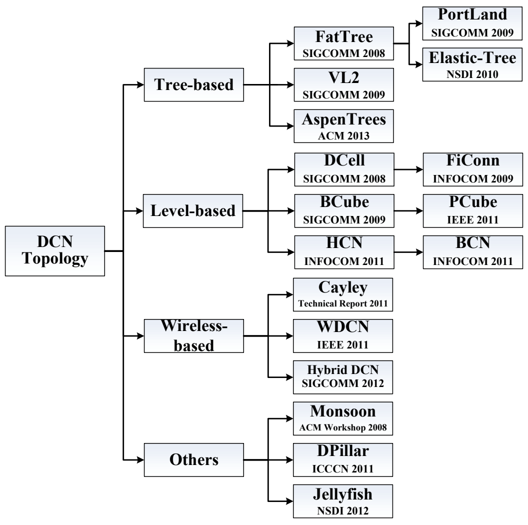

Fig. 2: Classification of data center topologies.

Considering the generalized Hypercube in Fig. 3, if we replace the nodes in the original hypercube with switches and insert one server into each link that connects two switches, resulting architecture meets the interconnection requirements.

2.2 60 GHz Wireless Technology

The 60 GHz wireless technology is now emerging to provide dense and fast connectivity at low cost. According to result of experiments with 60 GHz horn antenna(see Fig. 4), these technologies target a short range of 10 meters and directional links to compensate for path loss and high signal-to-noise ratio. It's shown that in a typical data center, line-of-sight 60 GHz links set up at rack height provide stable performance [8].

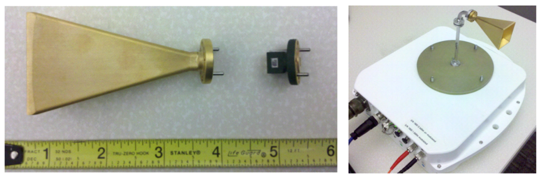

Fig. 4: HXI device, paired with a horn antenna.

As is mentioned above, we have designed a novel architecture for data center. Because a quantity amount of servers will placed into a rack,then a switch is put on the top of rack, we can equip every switch with an horn antenna to relieve hotspots in oversubscribed data center.

2.3 Transport Protocol for Data Centers

As wireless technology has introduced in our design, traditional Transport Control Protocol has to be modified to meet the needs. More importantly, the improved transport protocol is readily deployed to handle bursts and delay sensitive or throughput sensitive flows.Before moving on, have a look at TCP algorithms presented in recent years.

As seen in Fig. 5,improved transport protocol for data center from 2010 to 2014 fall into three categories, namely, implicit rate control. explicit rate control and flow scheduling. The first category includes ICDCT [27], DCTCP [6], D2TCP [28], HULL [29], L2DCT [30]; the second category consists of D3 [31], PDQ [32], and pFabric [33]; the last category only includes RepFlow [34]. Considering the wireless technology, we proposed our algorithm based on DCTCP.

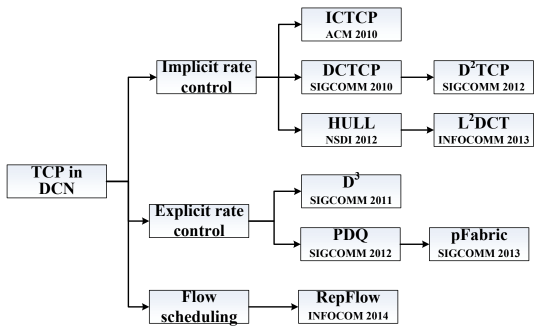

Fig. 5: Classification of Improved TCP for data center.

3 WIRELESS SWCUBE

3.1 Construction of Wireless SWCube

As is mentioned above, the construction of SWCube is based on the generalized hypercube.This architecture can be constructed logically as follows:

Constructing Hypercube(r,k): We denote a k-dimensional generalized hypercube by HyperCube(r,k), which means r nodes in every dimension for symmetry and regularity. A node W is represented by a k-tuple: $W = w_1 w_2 \cdots w_k$, where $0 < w_i \leq r_i - 1$, $\forall i = 1, 2, \cdots, k$. Two nodes are connected directly by a link if and only if their addresses differ at one bit. And Fig. 3 represent a HyperCube (4,2).

Replace all the nodes with switches: After step 1, we got the HyperCube(r,k). And then, replace all the nodes in the original generalized hypercube with switches. The unfinished structure is named SWCube because the SWitches form a generalized hypercube. And switches in SWCube can adopt the same addressing scheme for nodes in the original hypercube, where switch $W = w_1 w_2 \cdots w_k$.

Insert one server into each link: As SWCube is finished, insert one server into each link that connects two switches.As every server directly connects to two switches, it is uniquely identified by $V = (V^1, V^2)$, where $V^1 = v_1^1 v_2^1 \cdots v_k^1$ and $V^2 = v_1^2 v_2^2 \cdots v_k^2$ represent the two switches that the server directly connects to.

Place one antenna on the top of switch: After that,construction of SWCube(r,k) is finished, as shown in Fig. 6.Because a quantity amount of servers will placed into a rack,then a switch is put on the top of rack, we can equip every switch with an horn antenna. Fig. 7 represents the Wireless SWCube(4,2).

3.2 Properties of Wireless SWCube(r,k)

Since the construction of SWCube(r,k) is based on the hypercube, in the ith dimension there are r switches. The number of switches in an SWCube is $N_w = r^k$.Since switches along the same dimension form a complete graph, each switch connects to the other $r-1$ switches via a server along the ith dimension. Thus, the number

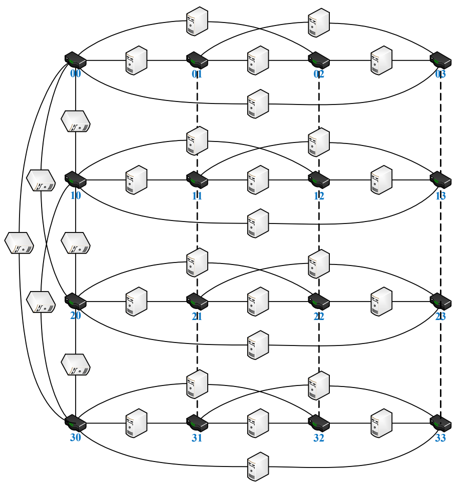

Fig. 6: Example of SWCube(r,k),where $r = 4, k = 2$.

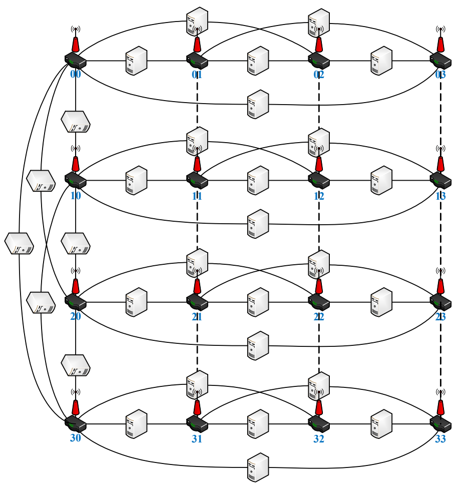

Fig. 7: Demonstration of Wireless SWCube(4,2).

of ports that are used in each switch is: $n = (r-1) \times k$. The of servers in an SWCube is actually the number of edges in the original generalized hypercube, which can be calculated as the number of switches N_w, times the switch port number n,divided by 2: $N_v = N_w \times n = kr^k(r - 1/2)$. Because every antenna is placed on top of switch, the number of antennas is the same as switches: $N_a = N_w = r^k$.Table 1 shows the results discussed above.

TABLE 1: Properties of Wireless SWCube(r,k)

Item	Properties of Wireless SWCube
Number of switches	$N_w = r^k$
Number of ports	$n = (r-1) \times k$
Number of servers	$N_v = kr^k(r - 1/2)$
Number of antennas	$N_a = r^k$

4 WIRELESS DEADLINE-AWARE TCP

The design of wireless deadline-aware TCP algorithm for our topology is based on the DCTCP algorithm.The goal of our design is to achieve high burst tolerance, low latency, and high throughput, with commodity shallow buffered switches. Standard TCP cuts its window size by a factor of 2 when it receives ECN(Explicit Congestion Notification) notification. In effect, TCP reacts to presence of congestion, not to its extent. while our design reacts to congestion in proportion to the extent of congestion. The wireless deadline aware TCP algorithm has five main components:

Flow scheduling at the switch: Wireless deadline-aware TCP employs a very simple queue management scheme with a queue threshold K. If queue occupancy Q is larger than K, the queue in the buffer are transported both by wire and wireless links, otherwise, they are transported only by wire links.

Considering the theoretical result in [6], the minimum of K is $K_{min} = (C \times RTT)/7$, where C is the capacity of a single wire link and RTT is the round-trip times.

Simple marking at the switch: The marking scheme is also very simple with a single parameter,the marking threshold W.If queue occupancy Q is larger than W, packets are marked with CE(Congestion Experienced) code point. Otherwise, it is not marked. Here W is calculated as follows:

$$W = (1 + \theta) \times K$$

where $\theta = \frac{Capacity of wireless link}{Capacity of wire link}$.

This scheme ensures that sources are quickly notified of the queue overshoot.

ECN-Echo at the Receiver: As the receiver ACK every packet, setting the ECN-Echo flag if and only if the packet has a marked CE codepoint, to accurately convey the exact sequence of marked packets back to the sender. Fig. 8 shows the ACK generation state machine.

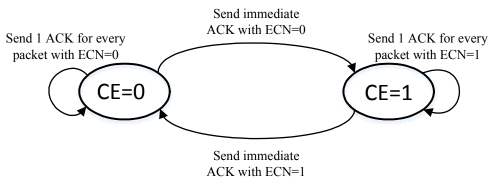
Send immediate ACK with ECN=0
Send 1 ACK for every packet with ECN=0
Send 1 ACK for every packet with ECN=1
CE=0
CE=1
Send immediate ACK with ECN=1

Fig. 8: Two state ACK generation state machine.

Controller at the Sender: As the sender maintains an estimate of the fraction of packets that are marked, called α, the extent of congestion updates once for every window of data (roughly one RTT) as follows:

$$\alpha = (1 - g) \times \alpha + g \times f$$

Here f is the fraction of packets that were marked with CE bits in the most recent window, and g is the weight given to new samples.Essentially, α close to 0 indicates low, and α close to 1 indicates high levels of congestion.

After that, we resize the congestion window $cwnd$ as follows:

$$\begin{aligned} cwnd &\leftarrow cwnd \times (1 - \alpha/2), if\ \alpha > 0 \\ cwnd &\leftarrow cwnd + 1, \qquad\quad if\ \alpha = 0 \end{aligned}$$

Deadline priority scheduling: We now define d as the deadline imminence factor as follows:

$$d = \frac{T_C}{D}$$

where T_C is time needed to complete transmitting and D is time remaining until deadline expires. Clearly,d larger than 1 indicates near deadline flow, and smaller than 1 indicates far deadline flow. Whenever a packet arrives to a port with a full buffer, if it has priority(d) less than or equal to the lowest priority packet in the buffer, it is dropped. Otherwise, the packet with the lowest priority is dropped to make room for the new packet.

5 CONCLUSIONS AND FUTURE WORK

This paper proposes and makes an analysis of a novel architecture for data center, Wireless Deadline-aware SWCube, with significant performance and practical advantages of high bandwidth, server expansive, burst tolerance and low cost.Further, we explore the wireless deadline-aware TCP algorithm for our wireless structure based on prior works. Considering the wireless link ands the deadline factor, this congestion avoidance algorithm meets the requirements for data center networks: low latency for short flow, high burst tolerance and high utilization for long flows.

Our future work will make the simulation on the Wireless Deadline-aware SWCube and the TCP algorithm. Another direction of our work is to make a comparison with the the architecture Fat-Tree.

REFERENCES

[1] S. Ghemawat, H. Gobioff, and S.-T. Leung. The google file system. In *Proc. SOSP*, NewYork, USA, 2003.
[2] K. Shvachko, H. Kuang, S. Radia, et al. The Hadoop Distributed File System. *Mass Storage Systems and Technologies*, 2010.
[3] F. Chang, J. Dean, S. Ghemawat, et al. Bigtable: A Distributed Storage System for Structured Data. *ACM Transactions on Computer Systems*, 2008.
[4] J. Dean, S. Ghemawat. MapReduce: Simplified Data Processing on Large Clusters. *Communications of the ACM*, 2008.
[5] M. Isard, M. Budiu, Y. Yu, et al. Dryad: Distributed Data-Parall Programs from Sequential Building Blocks. *ACM SIGOPS Operating Systems Review*, 2007.
[6] M. Alizadeh, A. Greenberg, D. A.Maltz, et al. Data Center TCP (DCTCP). In *Proc. of SIGCOMM*, New Dehli, India,2010.
[7] L. Bhuyan and D. Agrawal. Generalized hypercube and hyperbus structures for a computer network. *Computers, IEEE Transactions on*, 1984.
[8] D. Halperin, S. Kandula, J. Padhye, et al. Augmenting data center networks with multi-gigabit wireless links. In *Proc. IEEE SIGCOMM*, Toronto, Canada, 2011.
[9] D. Li, J. Wu, On the Design and Analysis of Data Center Network Architectures for Interconnecting Dual-Port Servers. In *Proc. IEEE INFOCOM*, Toronto, Canada, 2014.
[10] M. Al-Fares, A. Loukissas, and A. Vahdat. A scalable,commodity data center network architecture. In *Proc. of SIGCOMM*,Seatle, Washington, USA, 2008.
[11] R. Mysore, A. Pamboris, N. Farrington, et al. PortLand: A Scalable Fault-Tolerant Layer2 Data Center Network Fabric. In *Proc. of SIGCOMM*, Barcelona, Spain, 2009.
[12] A. Greenberg, N. Jain, S. Kandula, et al. VL2: A scalable and flexible data center network. In *Proc. of SIGCOMM*, Barcelona, Spain, 2009.
[13] B. Heller, S. Seetharaman, P. Mahadevan, et al. ElasticTree: Saving Energy in Data Center Networks. In *NSDI*, 2010.
[14] M. Walraed-Sullivan, A. Vahdat, K. Marzullo. Aspen Trees: Balancing Data Center Fault Tolerance, Scalability and Cost. In *Proc.ENET, ACM*, 2013.

[15] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu. DCell: A scalable and fault-tolerant network structure for data centers. In *Proc. of SIGCOMM*, Seatle, Washington, USA, 2008.

[16] D. Li, C. Guo, H. Wu, X. Zhang, and S. Lu. Ficonn: Using back port for server interconnection in data centers. In *Proc.IEEE INFOCOM*, Brazil, 2009.

[17] C. Guo, G. Lu, D. Li, H. Wu, et al. "BCube: A high performence,sever-centric network architecture for modular data centers," in *Proc.SIGCOMM*, Barcelona, Spain, 2009.

[18] L. Huang, Q. Jia, X. Wang, et al. PCube: Improving Power Efficiency in Data Center Networks. In *Cloud Computing,IEEE International Conference*, 2011.

[19] D. Guo, T. Chen, D. Li, Y. Liu, X. Liu, and G. Chen. BCN: Expansible Network Structures for Data Centers Using Hierarchical Compound Graphs. In *Proc.IEEE,INFOCOM*, Ontaria, Toronto, Canada, 2011.

[20] X. Liu, S. Yang, L. Guo, et al. Snowflake: A New-Type Network Structure of Data Center. *Chinese Journal of Computers*, 2011.

[21] J. Shin, E. G.Sirer, H. Weatherspoon, and D. Kirovski. On the Feasibility of Completely Wireless Datacenters. In *Technical Reports, Cornell University*,2011.

[22] Y. Cui, H. Wang, X. Cheng, et al. Wireless Data Center Networking. *Wireless Communications, IEEE*, 2011.

[23] X. Zhou, Z. Zhang, Y. Zhu, et al. Mirror mirror on the ceiling: flexible wireless links for data centers. In *Proc. of SIGCOMM*,New York, USA, 2012.

[24] A. Greenberg, P. Lahiri, D. A.Maltz, et al. Towards A Next Generation Data Center Architecture: Scalability and Commoditization. In *Proc.ACM workshop*, 2008.

[25] Y. Liao, D. Yin, and L. Gao. DPillar: Scalable Dual-Port Server Interconnection for Data Center Networks. *IEEE ICCCN*, Zurich, Switzerland, 2010.

[26] A. Singla, C. Hong, L. Popa, et al. Jellyfish: Networking Data Centers Randomly. In *Proc. USENIX conference on NSDI*, 2012.

[27] H. Wu, Z. Feng, C. Guo, and Y. Zhang. ICTCP: Incast Congestion Control for TCP in Data Center Networks. In *ACM CoNEXT*, 2010.

[28] B. Vamanan, J. Hasan, and T. N. Vijaykumar. Deadline-Aware Datacenter TCP(D2TCP). In *Proc. of SIGCOMM*, Helsinki, Finland, 2012.

[29] M. Alizadeh, A. Kabbani, T. Edsall, et al. Less is More: Trading a little Bandwidth for Ultra-Low Latency in the Data Center. In *NSDI*, 2012.

[30] A. Munir, I. A.Qazi, Z. A.Uzmi, et al. Minimizing flow completion times in data centers. In *Proc. IEEE INFOCOM*, Turin, Italy, 2013.

[31] C. Wilson, H. Ballani, T. Karagiannis, and A. Rowtron. Better never than late: meeting deadlines in datacenter networks. In *Proc. of SIGCOMM*, Toronto, Canada, 2011.

[32] C. Hong, M. Caesar, and P.B. Godfrey. Finishing Flows Quickly with Preemptive Scheduling. In *Proc. of SIGCOMM*, Helsinki, Finland, 2012.

[33] M. Alizadeh, S. Yang, M. Sharif, S. Katti, N. McKeown, B. Prabhakar, and S. Shenker. pFabric: Minimal Near-Optimal Datacenter Transport. In *Proc. of SIGCOMM*, Hong Kong, China, 2013.

[34] H.Xu, B. Li. RepFlow: Minimizing Flow Completion Times with Replicated Flows in Data Centers. In *Proc. IEEE INFOCOM*, Toronto, Canada, 2014.