© 2018 IJSRSET | Volume 4 | Issue 7 | Print ISSN: 2395-1990 | Online ISSN : 2394-4099
Themed Section : Engineering and Technology

Providing Good Expandability for the Networks by using Dual Port Server

R. Hamsaveni[1], A. Lohitha[2] , G. Sureshbabu[3]

[1]Assistant Professor, Department of ComputerApplications, Sri Venkateswara College of Engineering and Technology, Chittoor, Andhra Pradesh,India

[2]PG Scholar, Department of Computer Applications,Sri Venkateswara College of Engineering and Technology, Chittoor, Andhra Pradesh, India

[3]PG Scholar, Department of ComputerApplications,Sri Venkateswara College of Engineering and Technology, Chittoor, Andhra Pradesh, India

ABSTRACT

Designing an economical configuration for information centers which will deliver sufficient information measure and consistent latency performance to an oversized variety of servers has been a crucial and difficult downside. In existing system, an opportunist cooperation strategy for D2D transmission by exploiting the caching capability at the users to manage the interference among D2D links. we have a tendency to think about overlay inband D2D, divide the D2D users into clusters, and assign completely different frequency bands to cooperative and non-cooperative D2D links. to produce high chance for cooperative transmission, we have a tendency to introduce a caching policy. to maximise the network output, we have a tendency to together optimize the cluster size and information measure allocation, wherever the closed-form expression of the information measure allocation issue is obtained. . during this paper, we have a tendency to gift a completely unique server-centric information center configuration known as BCube connected crossbars (BCCCs), which may offer sensible network performance exploitation cheap trade goods off-theshelf switches and trade goods servers with solely 2 network interface card (NIC) ports. a major advantage of BCCC is its sensible expandability. once there's a desire for growth, we are able to simply add new servers and switches into the present BCCC with very little alteration of the present structure. Meanwhile, BCCC will accommodate an oversized variety of servers whereas keeping a awfully little network diameter. A fascinating property of BCCC is that its diameter will increase solely linearly to the network order (i.e., the amount of dimensions), that is superior to most of the present server-centric networks, like FiConn and BCN, whose diameters increase exponentially with network order. additionally, there square measure an expensive set of parallel methods with similar length between any try of servers in BCCC, that allows BCCC to not solely deliver ample information measure capability and predictable latency to finish hosts, however conjointly offer sleek performance degradation just in case of part failure. we have a tendency to conduct comprehensive comparisons between BCCC with alternative common server-centric network topologies, like FiConn and BCN. we conjointly propose a good addressing theme and routing algorithms for BCCC.

Keywords : BCube connected crossbars (BCCC), network interface card, topology, server-centric networks

I. INTRODUCTION

A prevalent server-driven server farm organize is BCube, which is a various leveled connect with numerous great properties, for example, low system measurement, productive steering et cetera. Be that as it may, BCube is constrained by its poor expandability, as growing a BCube organize, however not feasible,

causes huge equipment and human endeavors, on the grounds that the current system structure needs to experience noteworthy control and every one of the servers require more NIC ports for extension. Consequently, BCube just fits in a restricted scope of utilizations, for example, measured server farm arranges inside a transportation holder. Another disadvantage of BCube is that the equipment cost increments definitely with the quantity of system chains of command (or requests), in light of the fact that a BCube system of an extensive request requires each associated server to have numerous NIC ports. Be that as it may, the larger part of product servers in current market are outfitted with just two NICs.

In this paper, we think about a novel server-driven system topology for server farm correspondence, called BCube Connected Crossbars, indicated as BCCC for short, which consolidates the benefits of the previously mentioned server-driven systems however disposes of their disadvantages. BCCC can be built utilizing double port item servers and product switches, which makes it a financially savvy arrange answer for substantial scale server farms. Another extremely attractive element of BCCC is that it is effortlessly expandable, as growing a current BCCC system to a bigger size with a higher system arrange requires no change to the current system foundation or the quantity of ports of a server, which incredibly encourages server farm overhaul. Not at all like FiConn or BCN, the width of BCCC increments just directly to the system arrange, which implies that the correspondence dormancy between end has in BCCC is altogether lower than Ficonn and BCN, particularly in expansive frameworks. Likewise, each combine of servers in BCCC is associated by means of a rich arrangement of parallel ways with close equivalent length, through which BCCC can convey adequate system transfer speed, as well as can keep up unsurprising system execution and vigor against part disappointment.

ALGORITHM:

One-to-One Routing

In one-to-one routing, a single source is sending packets to a single destination. Suppose two servers A and A want to communicate. Let h(A, A) be the hamming distance of A and A , defined as the number of different digits between their addresses. According to the aforementioned construction procedure, we can see that the maximum hamming distance between any pair of servers in a BCCC(n, k) is k + 2. Note that a server has one-hop distance to all its neighbors. Thus, a path between a source and a destination usually includes multiple hops via intermediate servers. A route between the source and the destination can be found iteratively. At each iteration, we move to some intermediate server to correct one digit ai, $1 \le i \le k + 1$, in the source server's address to the corresponding digit in the destination server's address. Here, each iteration is divided into two steps. In step 1, in BCCC, in order to change ai to a i, a0 must be i – 1, or we can say a0 must be on the i th dimension. If not, denote the intermediate node in the current iteration as B, where bi = ai, then B must route to another server B first, where bj = b j , $1 \le j \le k + 1$ and b 0 = i – 1. For example, in Fig. 2, if we want to route from server 000 to server 101, where a2 = 0, a 2 = 1 and i = 2. As a0 = 0 = i–1, we cannot directly correct a2 to a 2 without going to another intermediate server first. Hence we need to change a0 to 1 and route to intermediate server 001. Then we can go to step 2. Step 2 is to change digit ai to a i. Like the example mentioned above, in step 2, server 001 routes to 101. It is easy to see that after at most k + 1 iterations, we can find a route between any source server and destination server.

```
Input: source A and destination A′, where
   A is a_{k+1}a_k a_{k-1} ... a_0 and A[i] = a_i
   A′ is a′_{k+1}a′_k a′_{k-1} ... a′_0 and A′[i] = a′_i, i ∈ [0, k+1]
   and Π which is the a permutation of [k+1, k, k−1, ..., 1]
Output: A list of intermediate servers of path from A to A′,
   path(A, A′)

BCCCRouting(A, A′, Π)
   path(A, A′) = {A};
   B = A;
   for (i = k+1; i > 0; i − −) // k+1 iterations
     if A[π_i] ≠ A′[π_i]
       B[0] = π_i − 1;
       if A ≠ B
         append B to path(A, A′); //step 1
       B′ = B;
       B′[π_i] = A′[π_i];
       B′ = B;
       append B′ to path(A, A′); //step 2
   if A′ ≠ B′
     append A′ to path(A, A′);
   return path;
```

BCCC routing algorithm to find a path from a single source to a single destination

One-to-All Routing

In one-to-all, or broadcast communication, one specific source server sends packets to all other servers in the network. Broadcast communication is required by many common network protocols, such as ARP, hence, an efficient broadcasting algorithm is necessary. In [9], a routing algorithm was proposed, which speeds up one-to-all traffic by dividing the file into (k + 1) parts, and sending each part using one independent edge-disjoint path. This algorithm enables certain speedup, however, it does not work in the scenarios such as real-time on-line communication, as dividing the object into parts may result in the disorder of packets received at the destinations, yet the order of these packets is critical. We now design a novel efficient broadcast routing algorithm for the proposed BCCC topology. This routing algorithm takes advantage of the hypercube-like structure of BCCC.

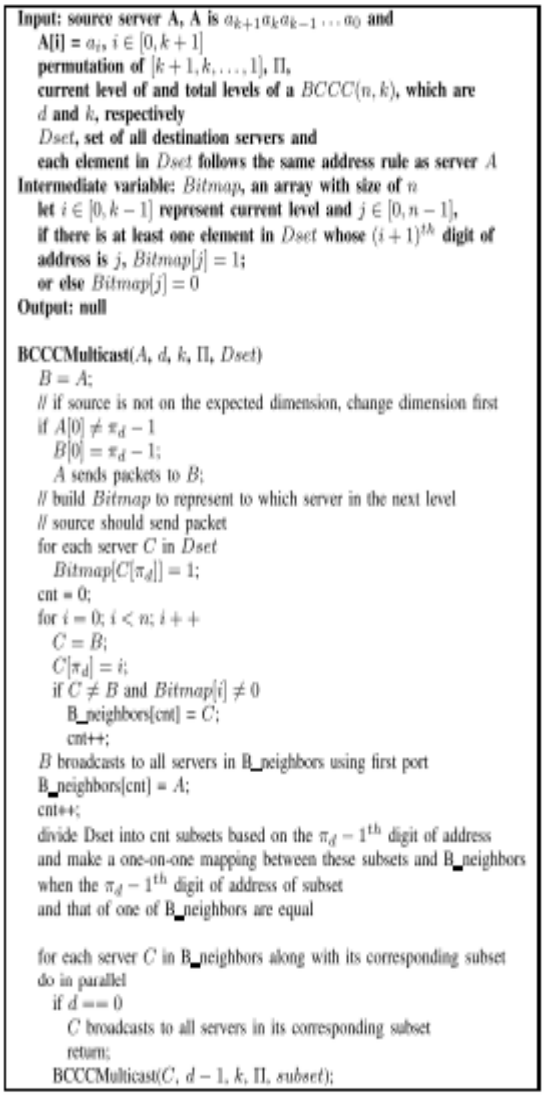
```
Input: source server A, A is a_{k+1}a_k a_{k-1} ... a_0 and
   A[i] = a_i, i ∈ [0, k+1]
   the permutation of [k+1, k, ..., 1], Π,
   current level of and the total levels of a BCCC(n, k),
   which are d and k, respectively
Output: null

BCCCBroadcast(A, d, k, Π)
   B = A;
   if A[0] ≠ π_d − 1
     B[0] = π_d − 1;
     A sends packets to B;
   cnt = 0;
   for i = 0; i < n; i++
     C = B;
     C[π_d] = i;
     if C ≠ B
       B_neighbors[cnt] = C;
       cnt++;
   B broadcasts to all servers in B_neighbors using first port
   B_neighbors[cnt] = A;
   for each server C in B_neighbors do in parallel
     if d == 0
       cnt = 0;
       for i = 0; i ≤ k; i++
         B = C;
         B[0] = i;
         if B ≠ C
           C_neighbors[cnt] = B;
           cnt++;
       C broadcasts to all servers in C_neighbors using second port
       return;
     BCCCBroadcast(C, d − 1, k, Π);
```

BCCC broadcasting algorithm to send packets from a single source server to all other servers in the network

One-to-Many Routing

In one-to-many or multicast communication, a specific source server is sending packets to many destination servers in the network. Existing multicast algorithms and protocols, such as IGMP and PIM designed for Internet, also apply to BCCC, however, they are usually destination driven algorithms or protocols. As discussed earlier, there exist multiple edgedisjoint parallel paths between any pair of servers in BCCC. The independent path selection in a destination driven algorithm will generate a multicast tree with many unnecessary intermediate links, which wastes bandwidth. Thus we need a more efficient multicast algorithm which takes advantages of BCCC.



Input: source server A, A is $a_{k+1}a_k a_{k-1}\ldots a_0$ and
A[i] = a_i, $i \in [0, k+1]$
permutation of $[k+1, k, \ldots, 1]$, Π,
current level of and total levels of a $BCCC(n,k)$, which are
d and k, respectively
$Dset$, set of all destination servers and
each element in $Dset$ follows the same address rule as server A
Intermediate variable: $Bitmap$, an array with size of n
let $i \in [0, k-1]$ represent current level and $j \in [0, n-1]$,
if there is at least one element in $Dset$ whose $(i+1)^{th}$ digit of address is j, $Bitmap[j] = 1$;
or else $Bitmap[j] = 0$
Output: null

BCCCMulticast(A, d, k, Π, $Dset$)
$B = A$;
// if source is not on the expected dimension, change dimension first
if $A[0] \neq \pi_d - 1$
$B[0] = \pi_d - 1$;
A sends packets to B;
// build $Bitmap$ to represent to which server in the next level
// source should send packet
for each server C in $Dset$
$Bitmap[C[\pi_d]] = 1$;
cnt = 0;
for $i = 0$; $i < n$; $i++$
$C = B$;
$C[\pi_d] = i$;
if $C \neq B$ and $Bitmap[i] \neq 0$
B_neighbors[cnt] = C;
cnt++;
B broadcasts to all servers in B_neighbors using first port
B_neighbors[cnt] = A;
cnt++;
divide Dset into cnt subsets based on the $\pi_d - 1^{th}$ digit of address
and make a one-on-one mapping between these subsets and B_neighbors
when the $\pi_d - 1^{th}$ digit of address of subset
and that of one of B_neighbors are equal

for each server C in B_neighbors along with its corresponding subset
do in parallel
if $d == 0$
C broadcasts to all servers in its corresponding subset
return;
BCCCMulticast(C, $d-1$, k, Π, $subset$);

BCCC multicasting algorithm to send packets from a single source server to a set of servers in the network

CONCLUSION:

In this paper, we propose a novel server-centric data center network topology, called BCCC, which is a recursive network structure. BCCC can be built using only dual-port servers regardless of network size, and its expansion requires little change to the existing network structure. These properties give BCCC a very good expandability. Also, there are a rich set of near-equal-length parallel paths between any pair of servers in BCCC, which enables BCCC to provision sufficient transmission bandwidth and have graceful performance degradation upon failure. Finally, the diameter of BCCC only increases linearly to the network order, which means that servers will enjoy low-latency transmission even in a large-size network. The combination of these desirable properties makes BCCC a promising networking choice for data center communication. We also present three efficient routing algorithms for one-toone, one-to-all and one-to-many communications respectively.

II. REFERENCES

[1]. L. N. Bhuyan and D. P. Agrawal, "Generalized hypercube and hyperbus structures for a computer network," IEEE Trans. Comput., vol. C-33, no. 4, pp. 323-333, Apr. 1984.

[2]. J. M. Gonzalez, V. Paxson, and N. Weaver, "Shunting: A hardware/ software architecture for flexible, high-performance network intrusion prevention," in Proc. 14th ACM Conf. Comput. Commun. Secur., 2007, pp. 139-149.

[3]. A. Singla, C.-Y. Hong, L. Popa, and P. B. Godfrey, "Jellyfish: Networking data centers randomly," in Proc. 9th USENIX Conf. NSDI, 2012, p. 17.

[4]. C. Guo et al., "DCell: A scalable and fault-tolerant network structure for data centers," in Proc. ACM SIGCOMM, Aug. 2008, pp. 75-86.

[5]. C. Guo et al., "BCube: A high performance, server-centric network architecture for modular data centers," in Proc. ACM SIGCOMM, Aug. 2009, pp. 63-74.

[6]. Z. Li, Z. Guo, and Y. Yang, "BCCC: An expandable network for data centers," in Proc. 10th ACM/IEEE Symp. ANCS, Oct. 2014, pp. 77-88.

[7]. D. Guo, "Expandable and cost-effective network structures for data centers using dual-port servers," IEEE Trans. Comput., vol. 62, no. 7, pp. 1303-1317, Jul. 2013.

[8]. D. Li, "FiConn: Using backup port for server interconnection in data centers," in Proc. IEEE INFOCOM, Apr. 2009, pp. 2276-2285.

[9]. D. Li and J. Wu, "On the design and analysis of data center network architectures for interconnecting dual-port servers," in Proc. IEEE INFOCOM, Apr-May 2014, pp. 1851-1859.

[10]. L. Kou, G. Markowsky, and L. Berman, "A fast algorithm for Steiner trees," Acta Inf., vol. 15, no. 2, pp. 141-145, 1981.

[11]. D. Li, Y. Li, J. Wu, S. Su, and J. Yu, "ESM: Efficient and scalable data center multicast routing," IEEE/ACM Trans. Netw., vol. 20, no. 3, pp. 944-955, Jun. 2012.

[12]. G. Lu, C. Guo, Y. Li, and Z. Zhou, "ServerSwitch: A programmable and high performance platform for data center networks," in Proc. 8th USENIX Conf. Netw. Syst. Design Implement. (NSDI), 2011, pp. 15-28.

[13]. G. Lu, Y. Shi, C. Guo, and Y. Zhang, "CAFE: A configurable packet forwarding engine for data center networks," in Proc. 2nd ACM SIGCOMM Workshop Program. Routers Extensible Services Tomorrow (PRESTO), 2009, pp. 25-30.

[14]. Y. Yang and G. M. Masson, "Nonblocking broadcast switching networks," IEEE Trans. Comput., vol. 40, no. 9, pp. 1005-1015, Sep. 1991.

[15]. Y. Yang and G. M. Masson, "The necessary conditions for Clos-type nonblocking multicast networks," IEEE Trans. Comput., vol. 48, no. 11, pp. 1214-1227, Nov. 1999.

[16]. Y. Yang, "A class of interconnection networks for multicasting," IEEE Trans. Comput., vol. 47, no. 8, pp. 899-906, Aug. 1998.

[17]. Y. Yang and J. Wang, "Wide-sense nonblocking Clos networks under packing strategy," IEEE Trans. Comput., vol. 48, no. 3, pp. 265-284, Mar. 1999.

[18]. Y. Yang and G. M. Masson, "Broadcast ring sandwich networks," IEEE Trans. Comput., vol. 44, no. 10, pp. 1169-1180, Oct. 1995.

[19]. M. Al-Fares, A. Loukissas, and A. Vahdat, "A scalable, commodity data center network architecture," in Proc. ACM SIGCOMM, 2008, pp. 63-74.

[20]. Z. Guo and Y. Yang, "On nonblocking multicast fat-tree data center networks with server redundancy," IEEE Trans. Comput., vol. 64, no. 4, pp. 1058-1073, Apr. 2015.

Author's Profile:

R.Hamsaveni working as an Assistant Prrofessor in Sri Venkateswara College of Engineering and Technology, Chittor, AP.

A.Lohitha the P.G degree from Sri Venkateswara College of Engineering and Technology, Chittoor, A.P in 2018.

G.Sureshbabu the P.G degree from Sri Venkateswara College of Engineering and Technology,Chittoor, A.P in 2018.