Topological and Structural Properties of Butterfly-de-Bruin: A Product Network of Wrap-Around Butterfly and de Bruin Networks

Osman Guzide, Ph.D.
Department of Computer Science, Mathematics, and Engineering
School of Natural Sciences and Mathematics
Shepherd University
Shepherdstown, WV25443
United States of America

Weidong Liao, Ph.D.
Department of Computer Science, Mathematics, and Engineering
School of Natural Sciences and Mathematics
Shepherd University
Shepherdstown, WV25443
United States of America

Abstract

This paper proposes and analyzes a novel interconnection network called Butterfly-de-Bruijn, which is a Product of wrap-around Butterfly and de Bruijn Networks. This new network, as a product of Butterfly (B_n) and de Bruijn (D_n) networks of degree n and denoted as B_nD_n, provides the connection among $n2^n2^n$ nodes with a diameter of |_3n/2_|, same as butterfly network, and a constant node degree of 8. We show that $B_nD_n$is symmetric, even though it is a product of symmetric and non-symmetric networks, and it contains 2^n distinct copies of B_n (wrap-around Butterfly Network). Also shown in this paper is that B_nD_n supports all cycle sub graphs except those of odd lengths when n is even and of odd lengths less than n.

Keywords: Interconnection Networks, Product Networks, Butterfly, de Bruijn

1. Introduction

As the common way to provide dedicated communication channels inside a high performance computer, the interconnection network was historically used to provide processor-to-processor and processor-to-memory connections. As the number of computational nodes and devices increases inside each computing system, the interconnection network has also been adopted to provide connections to I/O devices, memory units, cache modules and memory buffers, and processing cores and elements insider other computer systems. It therefore creates a demand for the development of new interconnection topologies with larger cardinalities and other desirable properties, such as being symmetric, with lower diameter, and of constant degree.

Many of these proposed topologies use modifications, extension, or enhancements to an existing topology, or the merge of two topologies so as to benefit from the good properties of both. For instance, the cube-connected cycle, as proposed by Preparata and Vuillemin (Preparata & Vuillemin, 1981), is a merger of a Hypercube and a ring, which as a result possesses properties ofa low diameter from the Hypercube and a low node degree from the ring. Hypercube is the product merged with the de Bruijnnet work (Ganesan & Pradhan, 1993) and the Butterfly network (Shi & Srimani, 1998). In both examples, the diameter of the resulting interconnection network topologies is the sum of diameters of the two child networks. Other examples of product interconnection networks include Scalable Twisted Hypercube (Alam & Kumar, 2011) and Torus-Butterfly Networks (Latifah & Kerami, Embedding on Torus-Butterfly Interconnection Network, 2012). The latter is the Cartesian product of Torus and Enhanced Butterfly.

Butterfly is a popular interconnection network used in parallel computing, and it is also used in peer-to-peer networks (Datar, 2002)(Tang, Hu, Zhang, Zhang, & Ai, 2003).The wrap-around Butterfly (Wagh & Guzide, 2005)network (denoted as B_n) has 2^n times more nodes than regular butterfly network and is of the same diameter. Each wrap-around Butterfly network B_n(degree 4) has $n2^n$ nodes each labeled with a pair (I,X), where I is an integer between 0 and n-1 and X is a binary vector of length n. There are four edges from each node (I,X) going to (I+1,X), $(I+1;X\oplus 2^I)$, (I-1;X) and $(I-1;X \oplus 2^{I-1})$ where $\oplus$denotes eXclusive OR (XOR) operation of two vectors. Due to its properties of being symmetric and of small diameter of only $\lfloor 3n/2 \rfloor$, the wrap-around butterfly network is very attractive in many applications. It is employed to implement mappings of many signal processing algorithms such as the fast Fourier transform as well as many basic structures such as cycles and trees. In addition to wrap-around Butterfly network, research has been done on other modified versions of Butterfly networks. Examples are Extended Butterfly Networks (Guzide & Wagh, Extended Butterfly Networks, 2005) and Enhanced Butterfly Networks (Guzide & Wagh, Enhanced Butterfly : A Cayley Graph with Node 5 Network, 2007).

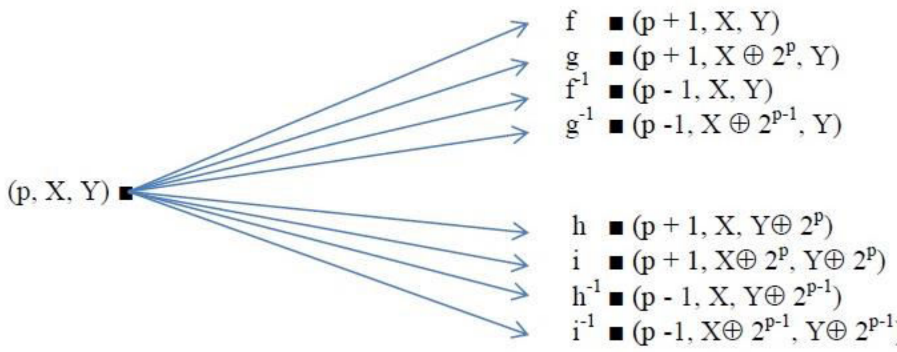
The de Bruijn Network is a hypercube network (of a fixed degree) that is derived from de Bruijn Graphs (Zhang & Lin, 1987)(Baker, 2011). It has been shown that a de Bruin graph(Baker, 2011), denoted as D_n, is a directed graph with d^n nodes labeled by n-tuples over a d-character alphabet. If d=2, D_n will have 2^n nodes that can be labeled with a bit representation (n bit) of the numbers 0, 1, …, 2^n-1.
In the graph, vertices are connected if the label of one end is the left or right shifted sequence of the other end, or it is the left or right-shifted sequence of the other end and differs correspondingly in the first or last bit. In this paper, we present a new Interconnection network based on the wrap-around Butterfly network and de Bruijn network. We show that by proper integration of the product of the wrap-around Butterfly Network B_n and de Bruijn Network D_n(2^n distinct copies of B_n), it is possible to derive a new symmetric network from the product of a symmetric network and a non-symmetric network, We name the resulting network as Butterfly-de-Bruijn and denote it as B_nD_n. A B_nD_n network of $n\times 2^n \times 2^n$ nodes will possess desirable properties such as a constant node degree of 8 and a diameter of B_nD_n is $\lfloor 3n/2 \rfloor$ (equal to that of B_n).

As shown in later section of this paper, $B_n D_n$ contains 2^n disjoint B_n copies as sub graphs, and can therefore support up to 2^nindependent algorithms designed for Butterfly networks. It also supports cycle and tree mappings much better than many other networks. We give here results for cycle sub graphs showing that in a B_nD_n network, when **n**is odd, cycles of all lengths up to $n\times 2^n \times 2^n$ (except odd lengths less than n) are sub graphs of B_nD_n. When n is an even, all even length cycles are sub graphs of B_nD_n. The rest of this paper is organized as follows. Section 2 describes elementary properties of a Butterfly-de-Bruijn networks as a product of Butterfly and de Bruijn Network. Section 3 presents and discusses other properties of the proposed product networks, including its routing strategy, diameter, and cycle sub graphs. We summarize this paper in Section 4.

2 Elementary Properties of Butterfly-de-Bruijn

Let Z_n denote the group of integers {0, 1, … ,n-1} with the operation of addition modulo n and Z_n^2, the group of binary vectors of length n under the operation of bit-by-bit modulo 2 addition of degree n $\geq$3, is defined as a graph onn2^n2^n nodes labeled by triples (p, X, Y) where p$\in$Zn and X,Y$\in Z_n^2$. As shown in Figure 1, a node in Butterfly-de-Bruijn (B_nD_n) is connected to the eight nodes. Let us denote the integer p and the vector components X and Y in a node label (p, X, Y) as the first, second and the third indices of the node respectively. The eight edges from the node of B_nD_n are labeled f,f^{-1}, g, g^{-1}, h, h^{-1}, i, and i^{-1}, as shown in Figure 1.

Note that edges of B_nD_n are bidirectional. In particular, for u, v $\in B_nD_n$, f(u) = v implies f^{-1}(v) = u,g(u) = v implies g^{-1}(v) = u, h(u) = v implies h^{-1}(v) = u, and i(u) = v implies i^{-1}(v) = u. Since every node of B_nD_n has a fixed node degree of 8, there are a total of $n\times 2^{n+2}\times 2^{n+1}$edges in B_nD_n.



f ■ (p + 1, X, Y)
g ■ (p + 1, X ⊕ 2^p, Y)
f^-1 ■ (p - 1, X, Y)
g^-1 ■ (p -1, X ⊕ 2^(p-1), Y)
(p, X, Y)
h ■ (p + 1, X, Y⊕ 2^p)
i ■ (p + 1, X⊕ 2^p, Y⊕ 2^p)
h^-1 ■ (p - 1, X, Y⊕ 2^(p-1))
i^-1 ■ (p -1, X⊕ 2^(p-1), Y⊕ 2^(p-1))

Figure 1: Connections from node (p, X, Y) in Butterfly-de-Bruijn Networks.

B_nD_n is a symmetric network. B_nD_n is the product of Butterfly and de Bruijn Networks and closely related to the popular wrap-around Butterfly B_n. Recall that $B_n$is a graph on $n\times2^n$ nodes, each with a label (p, X), where $p\in Z_n$ and $X \in Z_n^2$. By comparing the $B_n$with the above definition of B_nD_n, the following can be observed.

Theorem 1.B_nD_n contains 2^n disjoint copies of B_n sub graphs.

Proof. Partition the nodes of B_nD_n in 2^n sets based on the first and second indices. Denote the set of nodes in a partition where all nodes have the same last index Y by$B_nD_n(*, *, Y)$. To show that the sub graph on nodes $B_nD_n(*, *, Y)$ is isomorphic to B_n, define a mapping(p,X) : $B_nD_n(*,*,Y)^{\varphi}{}_{p:}B_nD_n(*, *, Y) \rightarrow B_n$ as $\varphi_p(p, X, Y) = (p,X)$. It is clear that φ_p (.) is a one-to-one onto mapping. Further, the edges within $B_nD_n(*,*,Y)$ are exactly mapped ontothe edges of B_n. For example, consider the edge$(p,X,Y) \rightarrow (p, X\oplus 2^p)$ of $B_nD_n(*,*,Y)$. Byusing the mapping φ_p (.), this edge translates to theedge $(p, X\oplus 2^p)\rightarrow (p,X)$ of B_n. Therefore, index Y has 2^n member of B_n.

Theorem 1 implies that 2^n instances of any algorithm designed to run on B_n networks can berun on B_nD_n networks without suffering any performance degradation. In addition, these instances can exchange information using the additional links present in B_nD_n networks. This structure also suggests a possibility of being able to map other algorithms on B_nD_n. In Section 4 we can see how to exploit the relationship between B_nD_n and B_n to develop cycle mappings on B_nD_n.

3. Analysis and Other Properties of $B_n D_n$ Networks

3.1 Routing Strategy

Routing strategy and diameter are important properties of any interconnection network. For the sake of efficient implementation of parallel algorithms, it is desirable to have a low diameter and a good routing strategy. Herein we provide an algorithm and strategy to obtain paths between nodes of B_nD_n networks.

Simple Path Algorithm to Go from (p_1, X_1, Y_1) **to**(p_2,X_2, Y_2) **in**$\lfloor 3n/2 \rfloor$**Hops**.

Step 1: Increase the second and third indices cyclically until the second and third indices becomes X_2 and Y_2 respectively.

For $p = p_1$ to p_k, $k=p_1-1$.
Let the current node be (p_1,X, Y).
If X and X_2 match and Y and Y_2 match in the (p)-th bit, then use edge f or g or h and go to node $(p + 1, X,Y)$ or $(p + 1, X\oplus 2^p,Y)$ or $(p + 1, X,Y\oplus 2^p)$ or $(p + 1,X\oplus 2^p , X\oplus 2^p)$.

Step 2: Increase the first index cyclically to p_2.

Let the current node be (p_k, X_2, Y_2).
If $-(n/2) \leq(p_2-p_k) \leq (n/2)$, then travel along edges f till the second index becomes p_2. Otherwise, continue along edges f^{-1} till the second index becomes p_2.

The correctness of the above path algorithm can be shown as follows. Note that in steps 1 of the algorithm, the first index increases by 1 at every hop. Thus in each hop, they can modify a different bit of the second and third indices. Together these steps use n hops and can therefore modify all then bits of the second and third indexes to make it X_2 and Y_2. Clearly, within these first n hops, the first index will become p_2 since $((p_2- p_1) \bmod n) < n$.

Finally, after the second step of the algorithm, the maximum distance between the first index and p_2 is at most |_n/2_|since one cango cyclically to approach p_2 from either direction by using edges forf^{-1}. Thus the last step of the algorithm uses at most |_n/2_|hops. Consequently, this algorithm provides a path of length at most |_3n/2_|between any pair of nodes in B_nD_n. To illustrate this algorithm, consider the path from node (4, 111111, 111111) to node (2, 000000, 000000) in B_6D_6.According to step 1 of the path algorithm, one would use 5 hops along edges g or f as follows:

(4. 111111, 111111) → (5, 101111, 101111) → (0, 001111, 001111) →(1, 001110, 001110) → (2,001100, 001100)→ (3,001000, 001000)→ (4,000000, 000000).

The step 2 of the algorithm then suggests that we should use 2 more hops along edge f^{-1}to modify the successive bits of the first index to match the destination. These two hops are as follows:

(4,000000, 000000) → (3,000000, 000000) → (2,000000, 000000).

Since in the present case one needs to change the first index from 4 to 2, using edges f^{-1} is prudent. These last two hops are:

(4,000000, 000000) → (3,000000, 000000) → (2,000000, 000000).

It should be noted that the above algorithm may not give an optimal path between the two nodes. But it is a simple algorithm and suffices to specify the diameter of B_nD_n as the following theorem illustrates.

3.2 Diameter of Butterfly-de-Bruijn

Theorem 2 (Diameter of B_nD_n). Diameter of Butterfly-de-Bruijn$B_nD_n$is|_3n/2_|.

Proof. As shown by the path algorithm given above, a path of length at most |_3n/2_| exists between any pair of nodes in B_nD_n. Therefore, to prove the theorem we merely have to show that |_3n/2_| is also the lower bound on the diameter. Following Theorem 1,we know that B_nD_n contains n copies of B_n sub graphs. Consider two nodes of B_nD_n which lie in the same copy of a B_n sub graph. It is obvious that the shortest path between the nodes uses only the edges of that sub graph. The distance between these two points in B_nD_n is the same as the distance between the corresponding points of the graph B_n. Thus the diameter of B_nD_n cannot be more than the diameter |_3n/2_| of B_n.

Theorem 2shows that even though the Product of the wrap-around Butterfly Network and de Bruijn B_nD_n has 2^n times as many nodes as a wrap-around Butterfly B_n, its diameter is the same as that of B_n. It is interesting to note that the path algorithm presented in this section uses edges f, g, h and i only. Therefore if one were to construct a directed graph B_nD_n which uses only these four edges, then the node degree would drop to 4, but the diameter would remain unchanged at |_3n/2_|.

3.3 Cycle Sub graphs

As indicated in **Theorem 1** as shown in Section 2, $B_n D_n$ contains 2^n disjoint copies of wrap-around Butterfly networks B_n. We use this fact to obtain larger cycle sub graphs of $B_n D_n$ by merging smaller cycle sub graphs located in these copies. To facilitate this, we first restate the following result from [6] that relates to the cycle sub graphs of Butterflies.

Theorem 3 (Cycle Sub graphs of B_n) [6]. Cycles of all lengths L are sub graphs of B_n except when:

a. odd L when n is even.
b. odd L less than n.
c. $L = 6$ when $n = 5$ or $n \geq 7$.
d. $L = 10$ when $n = 7$, $n = 9$ or $n \geq 11$.

This paper does not intend to discuss the designs of cycles in B_n. It is sufficient to note that for lengths smaller than 4×n, these cycles are generated using a template given in [6].For larger lengths, one first obtains a cycle sub graph of length L' such that n|L' and L – L' is a small eve number that is smaller than or equal to 2(n-1). One can then attach up to (n-1) additional pairs of nodes to this cycle to get the length L cycle. Cycle sub graphs of length L' are obtained by judiciously picking edges h and i (see Fig. 1 for the edge naming convention) to form the cycle. Recall that B_nD_n contains 2^n distinct copies of B_n, each made up of those nodes of $B_n D_n$ which have the same first index. By identifying cycle sub graphs in these copies of B_n and merging them together, one can get the desired cycle sub graphs of B_nD_n.

The following two lemmas can be used as guidelines to carry out this merging.

Lemma 1 (Cycle Merging using Edges g). Given a node pair u, v ϵB_nD_n connected by an h-edge, i.e.,h(u) = v, there exists another node pair z,w ϵB_nD_n, also connected by an h-edge, i.e. w = h(z) such that g(u) = z and g(v) = w. Further, the four nodes v, u,w, and z are distinct.

Proof: Let u = (p, X, Y). Then, v = h(u) = $(p+1, X, Y\oplus 2^p)$. One can verify that the nodes z = (p + 1, X, Y)and w = h(z) = (p +1+1,X,Y) satisfy the required conditions of the lemma.

Figure 2 illustrates the connections specified in Lemma 1. Note that nodes u and v in this figure belong to the same copy of B_n (they have the same second index) and w and z to another copy. One can relate node pairs in different copies of B_n by h edges as well. This is given in Lemma 2.

Lemma 2 (Cycle Merging using Edges f). Given a node pair u, v ϵB_nD_n connected by an h-edge, i.e.,h(u) = v, there exists another node pair z,wϵB_nD_n, also connected by an h-edge, i.e. w = h(z) such that f(u) = z and f(v) = w. Further, the four nodes v, u,w, and z are distinct.

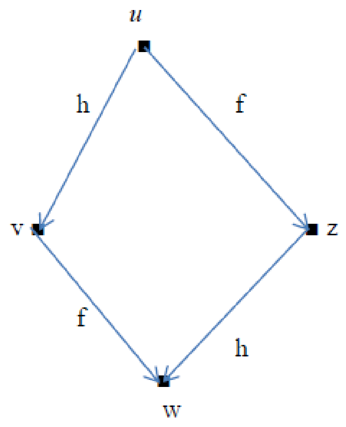


Figure 2: The connection of four nodes in B_nD_n.

Proof: One can verify that the nodes u = (p, X.Y), v = $(p+1,X, Y\oplus 2^p)$, z = $(p + 1, X,Y\oplus 2^p)$ andw = $(p+1+1, X, Y\oplus 2^p)$ satisfy the required conditions of the lemma.

Before we identify the cycle sub graphs of B_nD_n, now let us first present a result that imposes a fundamental limit on the cycle sub graphs of B_nD_n.

Theorem 4 (Impossible Cycle Sub graphs of B_nD_n). Cycles of the following lengths L are never sub graphs of B_nD_n:

a. Odd L when n is even.
b. Odd L less than n.

Proof: For an even n, partition the nodes of B_nD_n into two sets based upon on whether the sum of the first two indices of a node is even or odd. Once can see clearly that all B_nD_n edges go only between these sets an, B_nD_n is a bipartite graph for even n and therefore cannot support odd length cycle sub graphs. Now consider a length L, L < n, cycle sub graph of B_nD_n. Since L < n, there exists an integer k,$0 \leq k < n$ such that no node on the cycle has the form(n-k-2,X,Y). Replace each node (p, X,Y) of the cycle by node ((p-k -1) mod n, X,Y).

Note that this does not change the cycle connectivity. The new cycle of the same length L will not haveany node whose first is n - 1. Consequently, this new cycle will not use any wrap-around edges (i.e., edges that go between nodes whose first or second index changes from n-1 to 0). Thus along this cycle, the sum of the first two indices of the nodes traversed alternates between odd and even. This implies that the cycle length L < n must be even.

We now state the main result of this section.

Theorem 5 (Cycle Sub graphs of B_nD_n). Cycles of all lengths, except those identified in Theorem 3, are sub graphs of B_nD_n.

Proof: Recall that nodes of B_nD_n can be partitioned into2^n distinct copies of Bn. By virtue of Theorem3, cycle sub graphs of all lengths up to $n2^n2^n$(except of lengths 6 and 10) specified in Theorem 5 exist inB_nD_n. Cycle sub graphs of length 6 and 10 may be directly constructed in B_nD_n as:

$(0,0,0) \rightarrow (1,0,0) \rightarrow (2; 2; 2) \rightarrow (1; 2; 3) \rightarrow (2,0,1) \rightarrow (1,0,1) \rightarrow (0; 0; 0)$.
$(0, 0,0) \rightarrow (1, 0,0) \rightarrow (2,0, 0) \rightarrow (1, 1,0) \rightarrow (2,3,2) \rightarrow (1, 3,2) \rightarrow (2, 3,2) \rightarrow (1,1,0) \rightarrow$
$(0, 1,0) \rightarrow (1,1,1) \rightarrow (0; 0; 0)$.

To obtain cycle sub graphs of lengths greater than $n2^n$,one may first design cycle sub graphs on multiple copies of B_n, and then merge them using Lemma 1 or Lemma2. Recall that the cycles in each copy of B_n use only edges.

The merging process using Lemma 2 is illustrated in Figure 3.

It is evident from Figure 3 that the merging of cycles in p-th and (p+1)-th copies of B_n is possible if there exists an edge $(p, X,Y) \rightarrow (p+ 1,X, Y\oplus 2^p)$ in the first cycle and an edge $(p + 1,X,Y) \rightarrow (p + 1+ 1, X,Y\oplus 2^p)$ in the second cycle. However, the index Y of all the nodes in a cycle maybe incremented by the same amount without destroying the cycle connectivity. Therefore the only requirement for merging the cycles in the two copies of B_n is the existence of an edge $(p, X, Y) \rightarrow (p+1, X ,Y\oplus 2^p)$in the first cycle and an edge $(p + 1, X , Y\oplus 2^p) \rightarrow (p+1+1,X , Y 2^p)$ in the second cycle for arbitrary Y_1 and Y_2. If one of these cycles has a length of at least 2^n-1, then this requirement can be met if the other cycle has an i edge. If the other cycle has only h edges, then one may use Lemma 1 in place of Lemma 2 in this proof. Thus cycles of all possible lengths $3 \leq L \leq n2^n2^n$are sub graphs of B_nD_n.

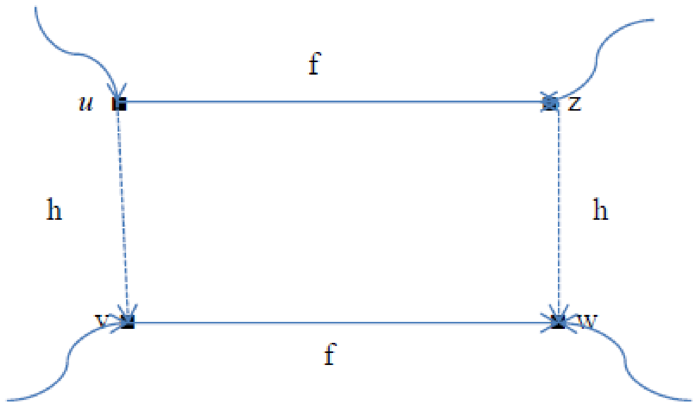


Figure 3: Merging cycles in two B_n sub graphs by removing the f-edges and adding h-edges between point su = (p, X, Y), v = (p+1, X, Y⊕ 2p), z = (p+1, X, Y) and w = (p + 1+1, X , Y⊕ 2p).

4. Conclusions

This paper describes and discusses B_nD_n, a product network of Butterfly (Bn) and de Bruijn (Dn) networks of degree n. B_nD_n contains 2^n distinct copies of B_n and therefore can run 2^n different algorithms designed for Butterfly networks without performance slowdown. The interconnection between these copies preserves all the desirable properties of the Butterfly network. The novel product network B_nD_n is a symmetric network with $n2^n2^n$ nodes and a constant node degree. It has a diameter equal to that of B_n, i.e.,|_3n/2_|. We have obtained a comprehensive solution to the problem of cycle sub graphs of B_nD_n. We have shown that B_nD_n does not have odd length cycles of lengths less than n. When n is odd, all cycles of length larger than n are sub graphs of B_nD_n. When n is even, all even length cycles are supported. It is instructive to compare the properties of B_nD_n with those of the Hyper-Butterfly (Shi & Srimani, 1998). The Hyper-Butterfly H_m is obtained by combining a Hypercube of degree m,H_m with B_n to get a network with $n2^{m+n}$nodes.Clearly, this may be much larger than the number of nodes $n2^n2^n$ of B_nD_n. On the other hand, B_nD_n has a constant node degree of 8 as compared with the no degree m + 4 of a Hyper-Butterfly. Similarly, the diameter of B_nD_n is only |_3n/2_|as compared with the diameter m+ |_3n/2_|of the hyper-Butterfly.

References

Alam, J., & Kumar, R. (2011). STH: A highly Scalable and Economical Topology for Massively Parallel Systems. Indian Journal of Science and Technology, 4(12), 1737-1748.

Baker, J. (2011). De Bruijn Graphs and Their Applications to Fault Tolerant Networks. Oregon: Oregon State University (Master Thesis).

Datar, M. (2002). Butterflies and Peer-to-Peer Networks. Proc. of the 10th European Symposium on Algorithms (pp. 310-322). BerlinHeidelberg: Springer.

Ganesan, E., & Pradhan, D. K. (1993, September). The Hyper-de Bruijn networks: Scalable versatile architecture. IEEE Trans. on Parallel and Distributed Systems, Vol. 4, pp. 962-978.

Guzide, O., & Wagh, M. D. (2005). Extended Butterfly Networks. Proc. of the The 18th Int. Conf. on Parallel and Distributed Computing Systems, (pp. 109-113). Las Vegas, NV.

Guzide, O., & Wagh, M. D. (2007). Enhanced Butterfly : A Cayley Graph with Node 5 Network. ISCA PDCS, (pp. 224-229).

Latifah, E., & Kerami, D. (2012). Embedding on Torus-Butterfly Interconnection Network. International Journal of Apllied Information Systems (IJAIS), Vol. 4. No. 12, 38-41.

Latifah, E., & Kerami, D. (2012). Structural Properties of Torus-Butterfly Interconnection Network. International Journal of Computer Applications (IJCA), 46(16), 31-35.

Preparata, F. P., & Vuillemin, J. (1981, May). The cube-connected cycles: a versatile network for parallel computation. Communications of the ACM, pp. 300-309.

Shi, W., & Srimani, P. K. (1998). Hyper Butterfly network: A scalable optimally faults tolerant architecture. Proceedings of the First Merged International Parallel Processing Symposium and Symposium on Parallel and Distributed Processing (pp. 732-736). Orlando, FL: IEEE Press.

Tang, Y., Hu, Z., Zhang, Y., Zhang, L., & Ai, C. (2003). A Practical Peer-Performance-Aware DHT. 2nd Int. Workshop on Agents and Peer-to-Peer Computing (pp. 193-200). Melbourne, Australia: Springer Berlin Heidelberg.

Wagh, M. D., & Guzide, O. (2005). Mapping Cycles and Trees on Wrap-Around Butterfly Graphs. SIAM Journal on Computing, 35, 741-765.

Wagh, M. D., Math, P., & Guzide, O. (2000). Cyclic-cubes and Wrap-around Butterflies. Information processing letters, 75(1), 25-27.

Zhang, F., & Lin, G. (1987). On the de Bruijn-Good Graphs. Acta Math. Sinica 30 (2), 195-205.