Extended Butterfly Networks

Osman Guzide
Department of CIS
Shepherd University
Shepherdstown, WV 25443

Meghanad D. Wagh
Department of ECE
Lehigh University
Bethlehem, PA 18015

Abstract

This paper defines a new network called the Extended Butterfly. The extended butterfly of degree n (XB_n) has $n^2 2^n$ nodes, diameter equal to $\lfloor 3n/2 \rfloor$ and a constant node degree of 8. XB_n is symmetric and contains n distinct copies of B_n. We also show that XB_n supports all cycle subgraphs except those of odd lengths when n is even and of odd lengths less than n.

1 Introduction

Recent demand for high speed and high throughput computational machines has led to the development of new interconnection topologies with larger cardinalities. Many of these topologies use modifications to an existing topology or merge two topologies so as to benefit from the good properties of both. For example, the cube-connected cycles proposed by Preparata and Vuillemin [1] is a merger of a Hypercube and a ring. It derives its low diameter from the Hypercube and low node degree from the ring. Similarly, Hypercube has also been merged with the de Bruijn network [2] and with the butterfly network [3]. The diameter of the network in both these cases is the sum of diameters of the two factor networks. This paper presents a new network based on the wrapped-around butterfly B_n which has n times more nodes than B_n but has the same diameter as B_n.

Butterfly architecture is a popular interconnection network used in parallel computing. It is also used in peer-to-peer networks [4, 5].

A degree n (wrap-around) butterfly network B_n 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 ExOR of two vectors. One can see that the last two of these edges are simply the first two edges in the reverse direction. Being a fixed node-degree network, B_n is easily scalable. The symmetry of B_n and its small diameter of only $\lfloor 3n/2 \rfloor$ makes it very attractive in many applications. B_n supports mappings of many signal processing algorithms such as the fast Fourier transform as well as many basic structures such as cycles and trees.

In this paper we show that by proper integration of n copies of B_n, one can get a new symmetric network, referred here as the *Extended Butterfly*, XB_n, with $n^2 2^n$ nodes and a constant node degree of 8. The diameter of XB_n is $\lfloor 3n/2 \rfloor$, equal to that of B_n. We also show that XB_n contains n disjoint B_n copies as subgraphs. It can therefore run up to n independent algorithms designed for butterfly networks. It also supports cycle and tree mappings much better than many other networks. We give here results for cycle subgraphs showing that when n is odd, cycles of all lengths up to $n^2 2^n$ (except odd lengths less than n) are subgraphs of XB_n. When n is even, all even lengths cycles are subgraphs of XB_n.

2 Elementary Properties

Let Z_n denote the group of integers $\{0, 1, \ldots, n-1\}$ with the operation of addition *modulo* n and Z_2^n, the group of binary vectors of length n under the operation of bit-by-bit *modulo* 2 addition. The Extended Butterfly, XB_n, of degree $n \geq 3$, is defined as a graph on $n^2 2^n$ nodes labeled by triples $(p,\ r,\ X)$ where $p, r \in Z_n$ and $X \in Z_2^n$. A node in XB_n is connected to the eight nodes shown in Fig. 1. Integers p, r and the vector X in a node label (p, r, X) are referred to as the first, second and the third indices of the node respectively. The eight edges from the node of XB_n are labeled g, g^{-1}, f, f^{-1}, h, h^{-1}, i, and i^{-1} as shown in Fig. 1.

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

Extended butterfly XB_n is closely related to the popular wrap-around butterfly B_n. Recall that B_n is

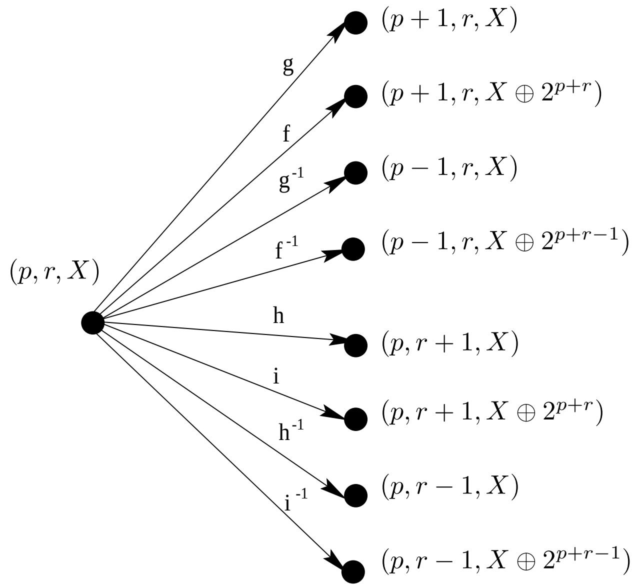

Figure 1: Connections from node $(p,\ r,\ X)$ in the Extended butterfly network.

a graph on $n2^n$ nodes, each with a label (p, X), where $i \in Z_n$ and $X \in Z_2^n$. By comparing the B_n with the above definition of XB_n, one can observe the following.

Theorem 1. XB_n contains n disjoint copies of B_n subgraphs.

Proof. Partition the nodes of XB_n in n sets based on the first index. Denote the set of nodes in a partition where all nodes have the same first index p by $XB_n(p, *, *)$. To show that the subgraph on nodes $XB_n(p, *, *)$ is isomorphic to B_n, define a mapping $\psi_p : XB_n(p, *, *) \rightarrow B_n$ as $\psi_p(p, r, X) = (r + p, X)$. Clearly, $\psi(\cdot)$ is a one-to-one onto mapping. Further, the edges within $XB_n(p, *, *)$ are exactly mapped onto the edges of B_n. For example, consider the edge $(p, r, X) \rightarrow (p, r+1, X \oplus 2^{p+r})$ of $XB_n(p, *, *)$. By using the mapping $\psi(\cdot)$, this edge translates to the edge $(r+p, X) \rightarrow (r+p+1, X \oplus 2^{p+r})$ of B_n. ∎

Theorem 1 immediately implies that n instances of any algorithm that is designed to run on B_n can be run on XB_n without any performance degradation. Further, these instances can exchange information using the additional links present in XB_n. This structure also suggests a possibility of being able to map other algorithms on XB_n. In Section 4 we exploits the relation between XB_n and B_n to develop cycle mappings on XB_n.

3 Routing and Diameter

Routing strategy and diameter are important properties of any interconnection network. A low diameter and a good routing strategy is important to efficient implementation of parallel algorithms. To this end, we first provide an algorithm to obtain paths between nodes of XB_n.

Simple path algorithm to go from (p_1, r_1, X_1) to (p_2, r_2, X_2) in $\lfloor 3n/2 \rfloor$ hops.

1. **(increase the first index cyclically to p_2.)**
   for $p = p_1$ to $p_2 - 1$
   Let the current node be (p, r_1, X).
   If X and X_2 match in the $(p+r_1)$-th bit, then use edge g and go to node $(p+1, r_1, X)$.
   else use edge f and go to node $(p+1, r_1, X \oplus 2^{p+r_1})$.

2. **(increase the second index cyclically till the third index becomes X_2)**
   for $r = r_1$ to $r_1 + p_1 - n - p_2 - 1$
   Let the current node be (p_2, r, X).
   If X and X_2 match in the (p_2+r)-th bit, then use edge h and go to node $(p_2, r+1, X)$.
   else use edge i and go to node $(p_2, r+1, X \oplus 2^{p_2+r})$.

3. **(increase or decrease second index to r_2)**
   Let the current node be (p_2, r, X_2).
   If $-(n/2) \leq (r_2 - r) \leq (n/2)$, then travel along edges h till the second index becomes r_2
   else, continue along edges h^{-1} till the second index becomes r_2.

Correctness of the above path algorithm is not difficult to prove. Note that in steps 1 and 2 of the algorithm, either the first or the second index increases by 1 at every hop. Thus in each hop, they can modify a different bit of the third index. Together these steps use n hops and can therefore modify all the n bits of the third index to make it X_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 second index and r_2 is at most $\lfloor n/2 \rfloor$ since one can go cyclically to approach r_2 from either directions by using edges h and h^{-1}. Thus the last step of the algorithm uses at most $\lfloor n/2 \rfloor$ hops. Consequently this algorithm provides a path of length at most $\lfloor 3n/2 \rfloor$ between any pair of nodes in XB_n.

To illustrate this algorithm, consider the path from node $(4, 0, 111111)$ to node $(2, 0, 000000)$ in XB_6. According to step 1 of the path algorithm, one would use 4 hops along edges g or f as follows:
$(4, 0, 111111) \rightarrow (5, 0, 101111) \rightarrow (0, 0, 001111) \rightarrow$

$(1, 0, 001110) \rightarrow (2, 0, 001100)$.
The step 2 of the algorithm then suggests that we should use 2 more hops along edges h or i to modify the successive bits of the third index to match the destination. These two hops are as follows:
$(2, 0, 001100) \rightarrow (2, 1, 001000) \rightarrow (2, 2, 00000)$.
Finally, the third step of the algorithm only adjusts the second index using hops along either h or h^{-1}. Since in the present case one needs to change the second index from 2 to 0, using edges h^{-1} is prudent. These last two hops are:
$(2, 2, 000000) \rightarrow (2, 1, 000000) \rightarrow (2, 0, 00000)$.

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 XB_n as the following theorem illustrates.

Theorem 2 (Diameter of XB_n). Diameter of the Extended Butterfly XB_n is $\lfloor 3n/2 \rfloor$.
Proof: As shown by the path algorithm given above, a path of length at most $\lfloor 3n/2 \rfloor$ exists between any pair of nodes in XB_n. Therefore to prove the theorem we merely have to show that $\lfloor 3n/2 \rfloor$ is also the lower bound on the diameter. Following Theorem 1, we know that XB_n contains n copies of B_n subgraphs. Consider two nodes of XB_n which lie in the same copy of a B_n subgraph. It is obvious that the shortest path between the nodes uses only the edges of that subgraph. The distance between these two points in XB_n is the same as the distance between the corresponding points of the graph B_n. Thus the diameter of XB_n cannot be less than the diameter $\lfloor 3n/2 \rfloor$ of B_n. ∎

Theorem 2 shows that even though the Extended Butterfly XB_n has 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. Thus if one were to construct a *directed* graph XB_n which uses only these four edges, then the node degree would drop to 4, but the diameter would remain unchanged at $\lfloor 3n/2 \rfloor$.

4 Cycle Subgraphs

As indicated in Section II, XB_n contains n disjoint copies of wrap-around butterfly B_n. We use this fact to obtain larger cycle subgraphs of XB_n by merging smaller cycle subgraphs located in these copies. To facilitate this, we first restate the following result from [6] that relates to the cycle subgraphs of butterflies.

Theorem 3 (Cycle subgraphs of B_n) [6]. Cycles of all lengths L are subgraphs 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 $n \geq 11$.

We do not discuss here designs of cycles in B_n. It is sufficient to note that for lengths smaller than $4n$, these cycles are generated using a template given in [6]. For larger lengths, one first obtains a cycle subgraph of length L' such that $n|L'$ and $L - L'$ is a small even number $\leq 2(n-1)$. One can then attach up to $(n-1)$ additional pairs of nodes to this cycle to get the length L cycle. Cycle subgraphs 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 XB_n contains n distinct copies of B_n, each made up of those nodes of XB_n which have the same first index. By identifying cycle subgraphs in these copies of B_n and merging them together, one can get the desired cycle subgraphs of XB_n. One needs the following two lemmas to carry out this merging.

Lemma 1 (Cycle merging using edges g). Given a node pair $u, v \in XB_n$ connected by an h-edge, i.e., $h(u) = v$, there exists another node pair $z, w \in XB_n$, 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 $u = (p, r, X)$. Then, $v = h(u) = (p, r+1, X)$. One can verify that the nodes $z = (p+1, r, X)$ and $w = h(z) = (p+1, r+1, X)$ satisfy the required conditions of the lemma. ∎

Fig. 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 first index) and w and z to another copy. One can relate node pairs in different copies of B_n by f edges as well. This is given in Lemma 2.

Lemma 2 (Cycle merging using edges f). Given a node pair $u, v \in XB_n$ connected by an i-edge, i.e., $i(u) = v$, there exists another node pair $z, w \in XB_n$, also connected by an i-edge, i.e. $w = i(z)$ such that $f(u) = z$ and $f(v) = w$. Further, the four nodes v, u, w, and z are distinct.

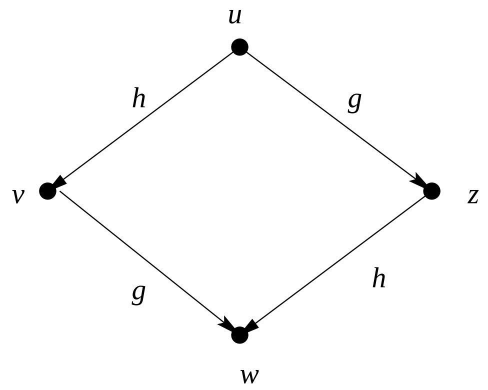

Figure 2: The connection of four nodes in XB_n.

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

Before we identify the cycle subgraphs of XB_n, we first present a result that imposes a fundamental limit on the cycle subgraphs of XB_n.

Theorem 4 (Impossible cycle subgraphs of XB_n). Cycles of the following lengths L are *never* subgraphs of XB_n:

a. odd L when n is even.

b. odd L less than n.

Proof: For an even n, partition the nodes of XB_n into two sets based upon on whether the sum of the first two indices of a node is even or odd. Clearly, all XB_n edges go only between these sets and no nodes in the same set are connected. Thus XB_n is a bipartite graph for even n and therefore cannot support odd length cycle subgraphs.

Now consider a length L, $L < n$, cycle subgraph of XB_n. Since $L < n$, there exists an integer k, $0 \leq k < n$ such that no node on the cycle has the form $(k, n-k-2, X)$. Replace each node (p, r, X) of the cycle by node $((p-k-1) \bmod n, (r+k+1) \bmod n, X)$. Note that this does not change the cycle connectivity. The new cycle of the same length L will not have any node whose first or second index 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 central result of this section.

Theorem 5 (Cycle subgraphs of XB_n). Cycles of all lengths, except those identified in Theorem 5, are subgraphs of XB_n.

Proof: Recall that nodes of XB_n can be partitioned into n distinct copies of B_n. By virtue of Theorem 3, cycle subgraphs of all lengths up to $n2^n$ (except of lengths 6 and 10) specified in Theorem 6 exist in XB_n. Cycle subgraphs of length 6 and 10 may be directly constructed in XB_n as:

$$(0,0,0) \to (0,1,1) \to (0,0,1) \to (1,0,0) \to (1,1,2) \to (0,1,0) \to (0,0,0).$$

$$(0,0,0) \to (1,0,0) \to (2,0,0) \to (2,1,0) \to (2,2,0) \to (1,2,8) \to (2,2,8) \to (1,2,0) \to (0,2,0) \to (0,1,0) \to (0,0,0).$$

To obtain cycle subgraphs of lengths greater than $n2^n$, one may first design cycle subgraphs on multiple copies of B_n, and then merge them using Lemma 1 or Lemma 2. Recall that the cycles in each copy of B_n use only h or i edges. The merging process using Lemma 2 is illustrated in Fig. 3.

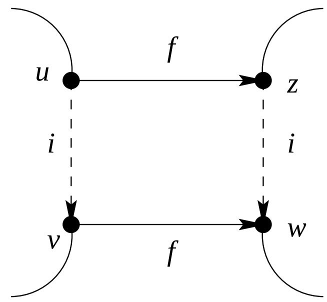

Figure 3: Merging cycles in two B_n subgraphs by removing the i-edges and adding f-edges between points $u = (p, r, X)$, $v = (p, r+1, X \oplus 2^{p+r})$, $z = (p+1, r, X \oplus 2^{p+r})$ and $w = (p+1, r+1, X \oplus 2^{p+r} \oplus 2^{p+r+1})$.

It should be clear from this figure that the merging of cycles in p-th and $p+1$-th copies of B_n is possible if there exists an edge $(p, r, X) \to (p, r+1, X \oplus 2^{p+r})$ in the first cycle and an edge $(p+1, r, X \oplus 2^{p+r}) \to (p+1, r+1, X \oplus 2^{p+r} \oplus 2^{p+r+1})$ in the second cycle. However, the index r of all the nodes in a cycle may be incremented by the same amount without destroying the cycle connectivity. Thus the only requirement for merging the cycles in the two copies of B_n is the existence of an edge $(p, r_1, X) \to (p, r_1+1, X \oplus 2^{p+r_1})$

in the first cycle and an edge $(p+1, r_2, X \oplus 2^{p+r_2}) \rightarrow (p+1, r_2+1, X \oplus 2^{p+r_2} \oplus 2^{p+r_2+1})$ in the second cycle for arbitrary r_1 and r_2. If one of these cycles has a length of at least $2^n - 1$, then this requirement can be easily 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 $3 \leq L \leq n^2 \times 2^n$ are subgraphs of XB_n. ∎

5 Conclusions

This paper describes XB_n, an extended butterfly network of degree n. XB_n contains n distinct copies of B_n and therefore can run n different algorithms designed for butterflies without any slowdown. The interconnection between these copies preserves all the good properties of the butterfly. XB_n is a symmetric network with $n^2 2^n$ nodes and a constant node degree. It has a diameter equal to that of B_n, i.e., $\lfloor 3n/2 \rfloor$. We have obtained a comprehensive solution to the problem of cycle subgraphs of XB_n. We have shown that XB_n does not have odd length cycles of lengths less than n. But when n is odd, all cycles of length larger than n are subgraphs of XB_n. When n is even, all even length cycles are supported.

It is instructive to compare the properties of XB_n with those of the Hyper-Butterfly [3]. Hyper-Butterfly is obtained by combining a Hypercube of degree m, H_m with B_n to get a network with $n 2^{m+n}$ nodes. Clearly, this may be much larger than the number of nodes $n^2 2^n$ of XB_n. On the other hand, XB_n has a constant node degree of 8 as compared with the node degree $m+4$ of a Hyper-Butterfly. Similarly, the diameter of XB_n is only $\lfloor 3n/2 \rfloor$ as compared with the diameter $m + \lfloor 3n/2 \rfloor$ of the hyper-Butterfly.

References

[1] F. P. Preparata and J. Vuillemin, "The cube-connected cycles: a versatile network for parallel computation," *Communications of the ACM*, vol. 24, pp. 300–309, May 1981.

[2] E. Ganesan and D. K. Pradhan, "The Hyper-deBruijn networks: Scalable versatile architecture," *IEEE Trans. on Parallel and Distributed Systems*, vol. 4, pp. 962–978, Sept 1993.

[3] W. Shi and P. K. Srimani, "Hyper butterfly network: A scalable optimally fault tolerant architecture," in *Int. Parallel Proc. Symp./Symp. on Parallel and Distributed Processing*, (Orlando, FL), pp. 732–736, 1998.

[4] M. Datar, "Butterflies and peer-to-peer networks," in *Proc. of the 10th Annual European Symp. on Algorithms*, pp. 310 – 322, 2002.

[5] Y. Tang, Z. Hu, Y. Zhang, L. Zhang, and C. Ai, "A practical peer-performance-aware DHT," in *2nd Int. Workshop on Agents and Peer-to-Peer Computing*, (Melbourne, Australia), July 2003.

[6] M. D. Wagh and O. Guzide, "Mapping cycles and trees on wrap-around butterfly graphs," *in review.*