

# Performance, Algorithmic, and Robustness Attributes of Perfect Difference Networks

Behrooz Parhami, *Fellow, IEEE*, and Mikhail A. Rakov

**Abstract**—Perfect difference networks (PDNs) that are based on the mathematical notion of perfect difference sets have been shown to comprise an asymptotically optimal method for connecting a number of nodes into a network with diameter 2. Justifications for, and mathematical underpinning of, PDNs appear in a companion paper. In this paper, we compare PDNs and some of their derivatives to interconnection networks with similar cost/performance, including certain generalized hypercubes and their hierarchical variants. Additionally, we discuss point-to-point and collective communication algorithms and derive a general emulation result that relates the performance of PDNs to that of complete networks as ideal benchmarks. We show that PDNs are quite robust, both with regard to node and link failures that can be tolerated and in terms of blandness (not having weak spots). In particular, we prove that the fault diameter of PDNs is no greater than 4. Finally, we study the complexity and scalability aspects of these networks, concluding that PDNs and their derivatives allow the construction of very low diameter networks close to any arbitrary desired size and that, in many respects, PDNs offer optimal performance and fault tolerance relative to their complexity or implementation cost.

**Index Terms**—Bipartite graph, chordal ring, diameter, emulation, fault tolerance, hyperstar, interconnection network, permutation routing, robust network, routing algorithm, scalability.



## 1 INTRODUCTION

DESIGNING interconnection architectures is an important and difficult task in the implementation of high-performance parallel and distributed systems [17]. The choice of the interconnection network may affect several characteristics of the final system, including implementation cost (node complexity, VLSI area, wiring density), performance, ease of programming, reliability, and scalability. Over the years, many different interconnection networks have been used in commercially available concurrent systems and numerous research prototypes [13], [15]; many more alternatives have been proposed and evaluated in theoretical studies or paper designs.

Mathematical models for interconnection networks have played important roles in understanding, synthesizing, and comparing a multitude of network architectures. Prime examples are found in ring networks [4], [5] and Cayley graphs [1], [12], which span a wide variety of networks and have been studied extensively by mathematicians, computer scientists, and computer design engineers. A demonstration showing that a number of symmetric, scalable, and economically viable interconnection networks are Cayley graphs [18], or belong to a class of derivative networks known as Cayley coset graphs [27], has led to renewed interest in these mathematical bases for producing symmetric and robust networks.

Other examples are offered by perfect difference networks (PDNs), and their derivatives such as multidimensional PDNs, which are based on the mathematical notion

of perfect difference sets [25]. In a companion paper [19], we have demonstrated that PDNs constitute worthy additions to the repertoire of computer system designers in offering additional design points that can be exploited by new and emerging technologies. In particular, the rich connectivity and small diameters of PDNs and related networks make them good candidates for wireless/optical network technologies [20], [21], [22], [23]. In this paper, we aim to complete the picture by considering complexity and scalability issues (Section 2), routing and performance (Section 3), collective communication (Section 4), robustness properties (Section 5), and comparison with other networks (Section 6).

Intermediate architectures between  $K_n$  (the  $n$ -node complete graph, which is a benchmark for high performance) and  $L_n$  or  $R_n$  (the  $n$ -node linear array or ring, which offers the ultimate in low cost and scalability) can be obtained in a variety of ways [6], [30]. The resulting networks provide trade-offs in cost (affected by parameters such as node degree  $d$ ) and performance (affected by network diameter  $D$  and bisection bandwidth  $B$ , among other factors). Fig. 1 is an abstract view of the design space for interconnection networks (including *perfect difference networks*, introduced in a companion paper [19] and reviewed briefly later in this section) in terms of diameter varying from the minimum 1 to the maximum  $n - 1$ . Cayley graphs exist in almost the entire span of Fig. 1, with examples ranging from ring and torus, through hypercube, star, and pancake networks, to the complete network. Even though networks near the left end of the spectrum in Fig. 1 are of little interest for wired connectivity due to cost and realizability factors, they can become practical and, thus, interesting, with wireless/optical links. Furthermore, richly connected networks of small to moderate size can form the

• B. Parhami is with the Department of Electrical and Computer Engineering, University of California Santa Barbara, CA 93106-9560. E-mail: parhami@ece.ucsb.edu.

• M.A. Rakov is with the Department of Computer Science, University of California, Santa Barbara, CA 93106-9560. E-mail: mikhrak@aol.com.

Manuscript received 12 Dec. 2003; revised 12 Aug. 2004; accepted 18 Oct. 2004; published online 22 June 2005.

For information on obtaining reprints of this article, please send e-mail to: tpd@computer.org, and reference IEEECS Log Number TPDS-0233-1203.

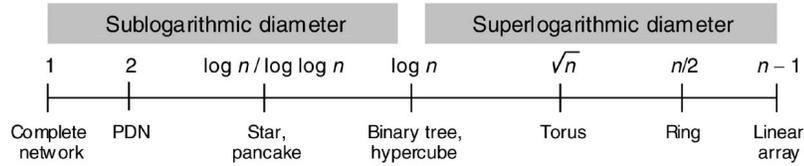


Fig. 1. The spectrum of interconnection networks in terms of diameter for size  $n$ .

clusters in multilevel networks and are thus of some interest even the wired case.

Given that the complete graph  $K_n$  (with diameter  $D = 1$ ) is impractical for large  $n$ , it is quite natural to consider the best topology for  $D = 2$ , the next most desirable network diameter. It is easily shown that a degree- $d$  undirected graph with  $D = 2$  can have no more than  $n = d^2 + 1$  nodes (Moore bound). *Perfect difference sets* (PDSs) provide the mathematical tools for achieving this optimum number of nodes, in an asymptotic manner, within the framework of perfect difference networks or PDNs. For an exposition of PDSs, their origins, and key properties, see our companion paper [19].

**Definition 1.** *Perfect difference set (PDS)*—A set  $\{s_0, s_1, \dots, s_\delta\}$  of  $\delta + 1$  integers having the property that their  $\delta^2 + \delta$  differences  $s_i - s_j, 0 \leq i \neq j \leq \delta$ , are congruent, modulo  $\delta^2 + \delta + 1$ , to the integers  $1, 2, \dots, \delta^2 + \delta$  in some order is a perfect difference set of order  $\delta$ . Any PDS can be converted to a normal form in which  $s_0 = 0, s_1 = 1$ , and  $1 < s_2 < s_3 < \dots < s_\delta$ . Henceforth, we deal only with normal-form PDSs of the form  $\{0, 1, s_2, \dots, s_\delta\}$ .

**Definition 2.** *Perfect difference network (PDN) based on the PDS  $\{0, 1, s_2, \dots, s_\delta\}$* —There are  $n = \delta^2 + \delta + 1$  nodes, numbered 0 to  $n - 1$ . Node  $i$  is connected to nodes  $i \pm 1$  and  $i \pm s_j \pmod{n}$ , for  $2 \leq j \leq \delta$ . Because all index expressions in this paper are evaluated modulo  $n$ , we will delete the qualifier “mod  $n$ ” throughout. The (basic) PDN just defined is an undirected chordal ring of degree  $d = 2\delta$  and diameter  $D = 2$  (Fig. 2a). If we use two node types, with host (switch) node  $i$  connected by

directed links to switch (host) node  $i(i), i + 1(i - 1)$ , and  $i + s_j(i - s_j)$ , a bipartite PDN with  $n$  host nodes and  $n$  switch nodes of in-degree and out-degree  $\delta + 1$  results (Fig. 2b).

## 2 COMPLEXITY AND SCALABILITY

In this section, we discuss some properties of PDNs that have a bearing on their realizability, implementation cost, and applicability to solving problems requiring varying levels of performance. Comparative evaluation relative to other networks is provided in Section 6.

The perfect difference network, with its  $O(n^{1/2})$  node degree in both the basic and bipartite forms, falls between the hypercube and complete graph in the design space of Fig. 1, offering performance close to the latter, at a much lower cost. If further cost reduction is desired, networks of smaller node degrees can be built based on the PDN concept. These networks fall in the space between hypercube and PDN in Fig. 1, offering somewhat lower performance than the latter at reduced cost, thus allowing cost-performance trade-offs in numerous configurations. For example, a 2D PDN (cross product of two PDNs) can reduce the node degree to  $O(n^{1/4})$  at the expense of increasing the diameter to 4. There will be more on this later in this section.

Let us first discuss the issue of scalability [7], a term that has been used in the parallel computing literature to mean at least three different things: 1) Possibility of building systems of any desired size using the same connectivity rules, 2) ease of expanding the system into the next larger

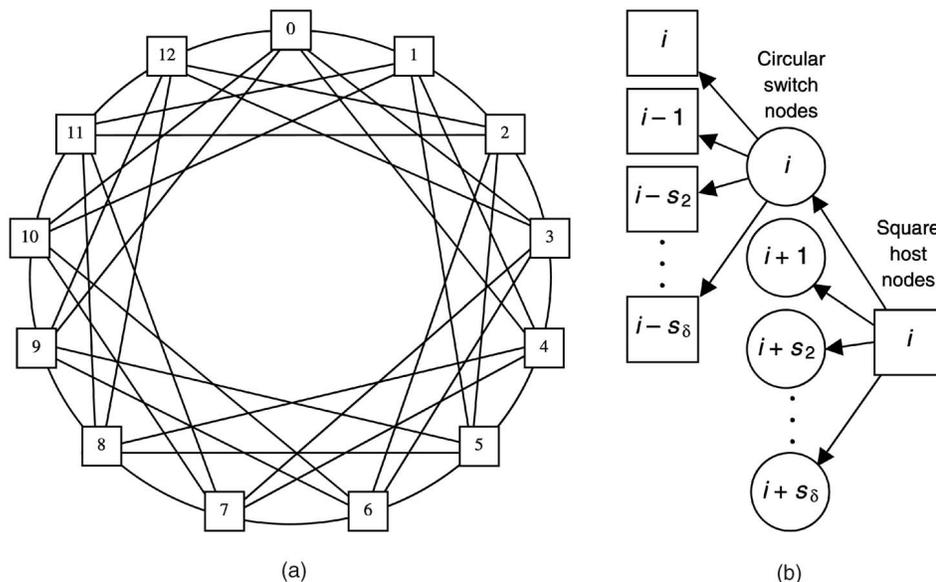


Fig. 2. The chordal ring structure of PDN with  $n = 13$  nodes based on the perfect difference set  $\{0, 1, 3, 9\}$ , and the strategy for deriving a  $2n$ -node bipartite PDN (with  $n$  circular and  $n$  square nodes) from an  $n$ -node PDN. (a) Basic PDN with  $n = 13$  nodes. (b) Deriving a bipartite PDN.

TABLE 1  
Scalability of PDN of Size  $n = \delta^2 + \delta + 1$ , and Comparison to Hypercube

$\delta$	$n$	PDN scalability		Hypercube	Hypercube scalability	
		Closest	Next size		Closest	Next size
2	7	0.300	0.625	8	0.333	0.778
3	13	0.235	0.500	16	0.333	0.882
4	21	0.192	0.409			
5	31	0.295	0.781	32	0.333	0.939
7	57	0.123	0.431	64	0.333	0.969
8	73	0.110	0.230			
9	91	0.188	0.446			
11	133	0.158	0.366	128	0.333	0.984
13	183	0.197	0.484			
16	273	0.059	0.120	256	0.333	0.992
17	307	0.108	0.237			
19	381	0.184	0.448			
23	553	0.081	0.175	512	0.333	0.996
25	651	0.075	0.161			
27	757	0.070	0.149			
29	871	0.065	0.139			
31	993	0.031	0.063	1024	0.333	0.998
32	1057	0.142	0.330			

configuration when computational requirements increase, and 3) constant, or very slowly rising, per-node cost as the network grows in size. PDNs are clearly not scalable in the third sense of the term. It is also fairly easily established that PDNs are not scalable in the second sense either. This is because the elements of a perfect difference set change for larger sizes. For example,  $\{0, 1, 3, 9\}$  is a PDS of order 3, but there is no PDS of order 4 that includes all four of its elements. So, whereas two smaller hypercubes can be combined into the next larger size by adding links to them (provided, of course, that the requisite channels are in place), no equivalent construction exists for going to the next larger PDN. We will see later that some composite networks based on PDNs do provide this type of scalability.

No regular network, other than the complete graph and certain chordal rings with fixed chord lengths, provides full scalability in the first sense of the term, that is, the ability to build a network of any desired size. Thus, in a practical sense, we say that a network is scalable if for any target size  $n_{\text{target}}$ , we can build a network with close to  $n_{\text{target}}$  nodes; the closer, the better. This type of scalability can be quantified by the percentage difference between the realizable size and the target size:  $|n_{\text{realizable}} - n_{\text{target}}|/n_{\text{target}}$ , with smaller values being more desirable. If the target number of nodes is viewed as a minimum requirement, then scalability must be derived based on the next larger network size, rather than based on the closest network size. Results of scalability analysis for PDNs of up to about 1K nodes are shown in Table 1. In the “Closest” column under “PDN scalability,” the worst-case relative difference to a closest network size is shown. For example, the first entry is  $(10 - 7)/10 = (13 - 10)/10 = 0.300$ , which occurs for  $n_{\text{target}} = 10$ . In the “Next size” column, the worst case occurs when the target network size is greater than an available size by 1; so, the first entry here is  $(13 - 8)/8 = 0.625$ . The corresponding values for hypercubes of comparable sizes are also shown in the rightmost two columns of Table 1, where the “Closest” value of 0.333 corresponds to

network sizes of the form  $2^a + 2^{a-1}$  and the “Next size” value, associated with  $n_{\text{target}} = 2^a + 1$ , tends to 1.

We see from Table 1 that scalability parameters of PDNs generally improve as the networks grow in size. This is no accident, as the trend can be established by noting that the  $i$ th prime number  $p_i$  is asymptotically equal to  $i \ln i$ , leading to the conclusion that  $(p_{i+1}^2 - p_i^2)/p_i^2 \approx 2/i$  tends to 0 for large  $i$ . This asymptotic justification is provided only to demonstrate that the perceived trend in Table 1 is in fact sustained; in practice, very large PDNs for which asymptotic analysis is applicable are not useful. Another way to assess the scalability is to note the trend of increasing separation between consecutive hypercubes in Table 1. Again, this trend can be justified by noting that asymptotically there are  $p/(\log_e p)$  prime numbers in the range  $[1, p]$ . So, even ignoring powers of primes, that can be used as values for  $\delta$ , there are some  $p/(\log_e p)$  PDNs of sizes up to  $p^2 + p + 1$ , whereas there are approximately  $2 \log_2 p$  hypercubes in the same range.

Scalability of networks can be improved (in all three senses of the term) by considering hierarchical or composite networks that use PDNs as building blocks. We are thus motivated to define multidimensional PDNs based on the notion of cross product networks [2].

**Definition 3.** *Multidimensional PDN*—Given the  $q$  PDNs  $H_0, H_1, \dots, H_{q-1}$  based on their respective PDSs of orders  $\delta_0, \delta_1, \dots, \delta_{q-1}$ , the cross product network  $H_{q-1} \times H_{q-2} \times \dots \times H_0$  is a  $q$ -dimensional PDN with nodes  $(x_{q-1}x_{q-2} \dots x_0)$ , where  $x_i$  belongs to the node set of  $H_i$ ,  $0 \leq i < q$ . When the  $H_i$  are identical, the product network  $H^q$  is a PDN-based power network.

All general properties of product networks immediately carry over to multidimensional PDNs as a special case. Some of the known results for product networks, summarized in

Table 1 of [3], lead to the following conclusions for  $H^q$ , where  $H$  is a PDN with  $n = \delta^2 + \delta + 1$  nodes. Corresponding results for general  $qD$  PDNs are easily formulated:

1. Number of nodes is  $n^q$ .
2. Node degree is  $2q\delta$ ; thus,  $d$  grows as the  $1/(2q)$ th power of the number of nodes.
3. Diameter is  $2q$ .
4. Average internode distance is  $q$  times that of  $H$ ; thus,  $\Delta = 2q\delta^2/n$  [19].

Other derivative results on routing and fault tolerance for multidimensional PDN based on the properties of product networks will be presented later in this paper.

When multidimensional PDNs are included in our assessment of scalability, the situation becomes much better. For example, just including power networks with identical components, adds the sizes 49, 169, 343, 441, and 961 to those listed in Table 1. Considering all product networks, including those with one or more components corresponding to the degenerate 3-node PDN based on the PDS  $\{0, 1\}$ , adds some 34 new sizes to those in Table 1, bringing the total number to 53 (51 of them below 1,000). The vast majority of these additions to available network sizes correspond to 2D PDNs, thus demonstrating the possibility of building very-low-diameter networks close to any arbitrary desired size.

Although VLSI layout of PDNs is not of much practical interest, we discuss it very briefly for completeness. The starting point for many VLSI layout studies is a collinear layout in which the  $n$  network nodes are placed along a straight line and tracks adjacent to the nodes are used to route the connections. Not surprisingly, the collinear layout of PDN requires an area that is intermediate between the  $O(n^2 \log n)$  area of hypercube's collinear layout and the  $O(n^4)$  area of the complete graph. This is because  $\min(s_i, n - s_i)$  parallel tracks are needed to accommodate all  $s_i$  links, leading to at most  $1 + s_2 + s_3 + \dots + s_\delta = O(n^{1.5})$  tracks and a total area of  $O(n^3)$  when the  $O(n^{0.5})$  width of each node is taken into account. The area requirements for multidimensional PDNs follow from VLSI layout relationships for product networks [3].

### 3 ROUTING AND PERFORMANCE

In this section, we discuss the problem of routing in PDNs. As interconnection networks differ in the way they accommodate message traffic, routing performance is a primary indicator of the overall benefits of a particular topology. In this section, we consider routing of a single point-to-point message and the capacity of PDNs to handle a random collection of such messages; routing for collective communication patterns typically encountered in parallel and distributed processing applications will be discussed in Section 4.

**Algorithm 1:** Oblivious shortest-path routing in a PDN—Oblivious routing from source node  $x$  to destination node  $y$  in  $H$  is tantamount to determining a unique intermediate node  $k$  such that  $k - x = s_i$  and  $k - y = s_j$ , for some pair of elements  $s_i$  and  $s_j$  in the PDS on which  $H$  is based. The problem thus reduces to the determination of  $s_i$ , that is, the first link to be traversed in the

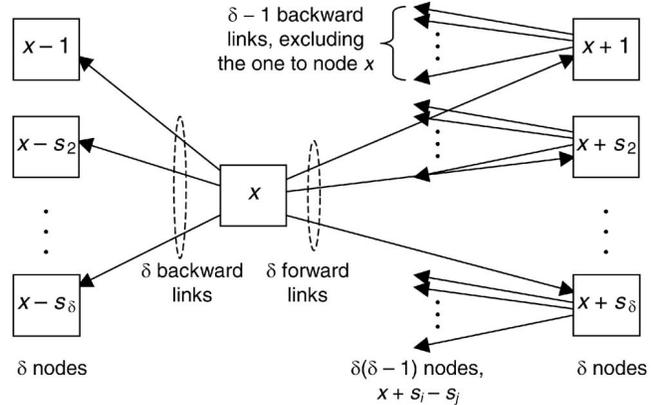


Fig. 3. Illustration of routing paths from node  $x$  to all other nodes.

forward direction (from  $x$  to  $k = x + s_i$ , with  $s_i$  possibly being 0) en route to node  $y$ , since the values of both  $k$  and  $s_j$  then follow. Fig. 3 depicts the routing paths from node  $x$  to all other nodes in the PDN. The value of  $s_i$  can be obtained based on table lookup or by means of online calculation of the differences between pairs of  $s_i$ .

The online calculation of differences has a complexity of at most  $O(n)$  steps, given that trying all  $\delta(\delta + 1)$  possible pairs of values for  $s_i$  and  $s_j$  is always applicable. Even though the complexity is not high, it is difficult to imagine routine use of such calculations for each message. It is more likely that such calculations are invoked when the primary path to the desired destination cannot be used. We will discuss adaptive and fault-tolerant routing shortly. As for the use of tables, every node needs a table of  $n - 1$  entries, with each entry specifying one of the  $\delta + 1$  forward links (including the one corresponding to  $s_0 = 0$ , or no forward movement). A link specification requires  $\lceil \log_2(\delta + 1) \rceil$  bits, resulting in a total table size of  $\delta(\delta + 1) \lceil \log_2(\delta + 1) \rceil \cong (n \log_2 n)/2$  bits. A table of this size is quite feasible for PDN configurations of practical interest.

To assess the distribution of messages on links for random internode message traffic due to Algorithm 1, we note that each link is used in the forward direction for sending messages to  $\delta$  different nodes. This is because each forward link leads directly to one node and indirectly, via the use of a backward link in the second step, to  $\delta - 1$  other nodes (Fig. 3). An identical argument establishes that each backward link is also used for messages coming from  $\delta$  different nodes. Thus, traffic tends to be balanced under random communication patterns, given that the  $n(n - 1)$  possible internode messages, traveling an average of  $2\delta^2/(n - 1)$  hops [19], for a total of  $2n\delta^2$  hops, when divided equally on  $n\delta$  links, leads to  $2\delta$  messages per link ( $\delta$  in each direction). The exact bisection width of PDN is unknown, but the fairly tight lower bound  $\lceil (\delta + 1)(n + 1)/4 \rceil$  for  $B$  [19] indicates that performance will be good. Note that the  $\Theta(n^{1.5})$  bisection width of PDN is intermediate between those of the hypercube (linear) and complete network (quadratic). The same relationship will hold between the routing performance of these networks.

Because nodes  $x$  and  $x + s_i - s_j$  ( $s_i, s_j \neq 0$ ) in a PDN are connected by two shortest paths via nodes  $x + s_i$  and  $x - s_j$ ,

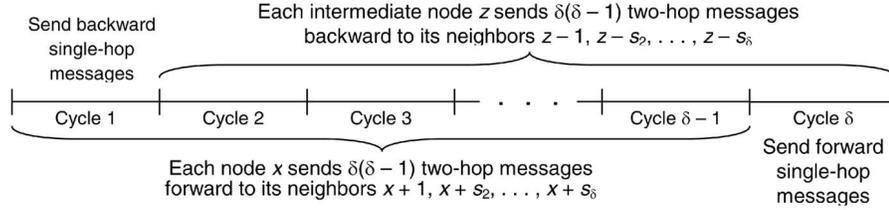


Fig. 4. Scheduling of pipelined message transmissions for optimal emulation of  $K_n$  by an  $n$ -node PDN.

the oblivious routing scheme of Algorithm 1 is needlessly restrictive.

**Algorithm 2:** Adaptive shortest-path routing in a PDN—This algorithm is similar to Algorithm 1, except that to send a message from node  $x$  to node  $y$  when the two nodes are not directly connected, either the forward link  $s_i$  or the backward link  $s_j$  is traversed first. One of the two node and edge-disjoint shortest paths can be chosen according to the availability of links (avoiding congested or faulty links/nodes) or by a traffic balancing scheme.

Having established the ease of optimal (oblivious or adaptive) routing and good communication performance for PDNs, it remains to be seen how parallel and distributed algorithms can be implemented to run efficiently on PDNs. Rather than develop such algorithms from scratch, we show how a PDN can emulate a complete network of the same size with a strictly optimal slowdown factor. This result demonstrates the emulative power of PDN and allows the porting of any parallel algorithm in view of the fact that a complete network can emulate any other network.

**Theorem 1.** *A PDN of order  $\delta$  can emulate a complete network of the same size with a slowdown factor of  $\delta$ . This emulation is strictly optimal.*

**Proof.** We first present an emulation with a slowdown factor  $2\delta$  that is easier to understand and can later be converted to the optimal emulation sought here. While analyzing random traffic distribution with the shortest-path routing of Algorithm 1 (see Fig. 3), we concluded that in sending messages from every node to every other node, every link in the PDN network will be used exactly  $\delta$  times in each direction. If we use these routing paths, the emulation of any communication step in the associated complete graph will be slowed down by a factor no greater than  $\delta$  due to congestion in the links and by a factor of at most 2 due to maximum dilation of 2. Note that the congestion-based slowdown factor of  $\delta$  holds for both half-duplex and full-duplex communication, provided the same assumption is made in both networks (the emulated complete graph and the emulating PDN). Given that the bisection widths of the complete graph and PDN are  $\Theta(n^2)$  and  $\Theta(n^{1.5})$ , respectively, this emulation is asymptotically optimal. We next derive a strictly optimal emulation. Because our emulation is based on Algorithm 1, each two-hop message takes a forward link followed by a backward link. These two steps can be pipelined according to the scheme depicted in Fig. 4. Referring to Figs. 3 and 4, the first  $\delta-1$  cycles are used by each node  $x$  to send its  $\delta(\delta-1)$  two-hop messages forward, with the last  $\delta-1$  cycles used to route the backward hop phase. The single-hop messages of node  $x$  are scheduled as

shown in Fig. 4 to complete the entire process in  $\delta$  cycles. Ensuring that conflicts will not delay the backward hop phase requires that in each step, an intermediate node  $z$  receives  $\delta$  messages from its backward neighbors that are headed to distinct nodes. This is easily accomplished by requiring a node  $x$  to send its forward messages to node  $z = x + s_i$  in cyclic order of the backward links to be used in the second phase:  $s_{i+1}, s_{i+2}, \dots, s_\delta, 1, s_2, \dots, s_{i-1}$ . Strict optimality of the  $\delta$ -step emulation follows from noting that there are  $n(n-1)$  messages to be sent over a network of average internode distance  $2\delta/(\delta+1)$  [19]. This requires that messages be sent over a total of  $2n(n-1)\delta/(\delta+1) = 2n\delta^2$  hops. The total number  $n\delta$  of links can accommodate  $2n\delta$  hops per time unit with full-duplex links; hence, a mandatory slowdown factor of  $\delta$ .  $\square$

The results presented in this section can be generalized to the case of multidimensional PDNs by means of the known properties of product networks [3]. For example, given a PDN  $H$ , optimal routing in  $H^2$  can be accomplished via generalized “row/column” routing where a message is first routed in the “row” PDN and then in the “column” PDN, or vice versa. Any sorting algorithm for a square mesh that uses row and column sorts as its basic components can be adapted to  $H^2$  by emulating a linear-array sorting algorithm on the row and column PDNs. Hence, our discussions of routing algorithms here and in Section 4 lead directly to a number of corresponding algorithms for  $H^2$  and other multidimensional PDNs with no additional effort.

We conclude this section with an observation about wormhole routing. Given the diameter of 2, wormhole routing is not attractive for PDNs. However, if wormhole routing is used, it is guaranteed to be deadlock-free given that at least one of several worms that conflict with each other at an intermediate node will be able to make progress toward its destination that is only one hop away. Wormhole routing is more attractive for multidimensional PDNs, but even there deadlock is impossible, as shown in the following theorem:

**Theorem 2.** *In a multidimensional PDN, a wormhole routing algorithm that uses shortest-path routing in each component and follows the generalized dimension-order routing scheme is guaranteed to be deadlock-free.*

**Proof.** We show that circular waiting is impossible. With dimension-order routing, a worm moves in a component PDN for one or two hops, before “turning” and starting to move in another component. Because distinct links are used in different components, and given that circular

waiting is impossible in each component, there is no circular waiting overall.  $\square$

#### 4 COLLECTIVE COMMUNICATION

Efficiency of realizing certain key collective communication patterns plays an important role in the usefulness of any interconnection network. The most important among these are one-to-all broadcasting (a source node sending a message to every other node), all-to-all broadcasting, complete exchange (every node sending a unique message to each of the other nodes), and a number of permutation routing patterns whereby each of the  $n$  nodes sends a message to a distinct node (so that  $n$  messages initially at their respective source nodes are permuted, each ending up at its destination node).

**Algorithm 3:** Broadcasting in a PDN—Broadcasting in  $H$  is a 2-phase process, given that the network diameter is 2:

- a. The initiating node  $x$  sends the broadcast message to node  $x + s_i$  for each nonzero member  $s_i$  of the PDS on which  $H$  is based. At the end of this phase, which involves  $\delta$  message transmissions,  $\delta + 1$  nodes are aware of the broadcast message.
- b. Each node  $y$  that already has the broadcast message (including the initiator) sends the message to nodes  $y - s_j$ , except for the node from which the broadcast message was received. In this phase,  $x$  sends  $\delta$  messages, while each of the  $\delta$  intermediate sources sends  $\delta - 1$  messages, for a total of  $\delta^2$  messages.

Broadcasting is thus completed in the minimum possible number  $\delta^2 + \delta$  of message transmissions. With the single-port communication model, where a node can send but one message in each time step, the total broadcast time is  $2\delta$  steps. All-port communication, on the other hand, leads to 2 time steps for broadcasting; this model is less realistic for networks with large node degrees and will not be considered further in this paper.

**Algorithm 4:** All-to-all broadcasting in a PDN—All-to-all broadcasting involves each node sending a message to all other nodes in the network; hence,  $n$  distinct messages must be sent, with each one going to  $n - 1$  destinations. Each node can follow Algorithm 3, with single-port communication, independently:

- a. The broadcast message of a node  $x$  is sent to all its neighbors  $x + s_i$  in  $\delta$  steps. These messages do not conflict with each other because the neighbors  $u + s_i$  and  $v + s_i$  for distinct nodes  $u$  and  $v$  are always distinct. At the end of this phase, each node has already received  $\delta$  of the expected  $\delta^2 + \delta$  broadcast messages.
- b. Each node  $y$  now sends its own message to all nodes  $y - s_j$ , and each of its received  $\delta$  messages to the same nodes, except the one from which the message originated. These steps require  $\delta + \delta(\delta - 1) = \delta^2$  message transmissions; again, there is never a conflict.

All-to-all broadcasting is thus completed in  $\delta^2 + \delta = n - 1$  message transmission steps, which is the minimum possible.

**Algorithm 5:** Complete exchange in a PDN—Complete exchange (also known as all-to-all personalized communication) involves each node sending a distinct message to each of the other nodes in the network; hence,  $n(n - 1)$  distinct messages must be transmitted, with each one having a single destination. Distinct messages must go from each node  $x$  to all nodes  $x + s_i - s_j$ . As  $s_i$  and  $s_j$  assume all values in the PDS, all distinct nodes are covered:

- a. Using  $2\delta$  message transmissions, each node  $x$  sends its messages to all destination nodes  $x + s_i$  and  $x - s_j$ , that is, nodes that can be reached in one hop. With all nodes using the same  $s_i$  or  $s_j$  in the same step, there is never a conflict.
- b. The remaining  $\delta(\delta - 1)$  destinations for each source must be reached in two steps. All nodes step through the possible  $s_i$  and  $s_j (s_i \neq s_j)$  values in unison, with node  $x$  sending a message to node  $x + s_i$ , requesting that it be forwarded to node  $x + s_i - s_j$ . This process requires  $2\delta(\delta - 1)$  message transmissions and again involves no conflict.

The algorithm requires  $2\delta^2$  message transmissions in all, which is optimal in view of each node sending  $n - 1$  messages and each message traveling the average internode distance of  $2\delta^2/(n - 1)$  hops [19]. The preceding algorithm and analysis assume that a node can send or receive a message in each time step, but not both at the same time. If simultaneous transmission and reception of messages is possible within a node, then the  $2\delta(\delta - 1)$  steps of phase b above can be reduced to  $\delta(\delta - 1) + 1$  through pipelining, leading to a complete exchange time of  $\delta^2 + \delta + 1 = n$  steps which is very close to the best possible value of  $n - 1$ . Note that all PDN links will carry the same number of messages in this algorithm; hence, message traffic is balanced. Note also that this result is not inconsistent with the  $\delta$ -step emulation of  $K_{n,r}$  proven in Theorem 2, as that result was based on an all-port communication model for both networks.

**Algorithm 6:** Shift permutation in a PDN—In a shift permutation, each node  $x$  sends a message to node  $x + c$ , where  $c$  is a constant. When  $c = \pm s_i$  for some  $i$ , shift permutation can be performed in one step with no conflict. In all other cases, two steps are needed, with all intermediate nodes  $x + s_i$ , and thus the links used, being distinct. Actually, a more general form of shift permutation in which the shift amount is  $+c$  for some nodes and  $-c$  for some others can be performed in the same number of steps, given that the forward-moving and backward-moving messages do not interfere with each other.

Many other common permutation routing problems can be solved by a sequence of (generalized) shift steps using Algorithm 6. The easiest way to develop the requisite algorithms is via emulating a hypercube. Consider for example the bit-reversal permutation among the subset 0 through  $2^{\lceil \log_2 n \rceil} - 1$  of nodes in a PDN. The hypercube algorithm can be emulated via  $\log_2 n$  shift steps in which the shift amounts are  $\pm 1, \pm 2, \dots, \pm 2^{\lceil \log_2 n \rceil - 1}$ . Similar algorithms can be devised for permutations corresponding to string reversal and matrix transposition.

We conclude this section by elaborating on collective communication in multidimensional PDNs. We describe the process for  $H^2$ , where  $H$  is a PDN, but generalization to  $H^q$  or an arbitrary  $qD$  PDN is straightforward. Optimal broadcasting in  $H^2$  follows from properties of product networks [3]: First broadcast in the row network and then in the column network, or vice versa, taking a total of four steps. All-to-all broadcasting is done by first performing all-to-all broadcasting in the row network, say, and then performing  $n$  all-to-all broadcasts in the column networks, leading to  $(n + 1)\delta^2$  transmission steps. Total overlap between row broadcasts and column broadcasts is possible with a slightly more sophisticated algorithm, leading to  $n\delta^2$  steps. For complete exchange, each node sends all of its messages for nodes in one column to the destination column's node in the same row. So, we need  $2n^2$  steps, or fewer with overlaps.

### 5 ROBUSTNESS PROPERTIES

Any computer system utilizing a large number of nodes and links must be robust if the failure of a very small subset of the many components is not to lead to total system crash. It is typical in robustness studies for interconnection networks to try to answer some or all of the following questions (posed in terms of the graph representing the interconnection pattern):

1. How many vertices can be removed without losing connectivity?
2. How many edges can be removed without losing connectivity?
3. How many vertex or edge-disjoint (shortest) paths exist between two nodes?
4. How is the network diameter affected if we remove some vertices?

Due to their mathematical construction and complete symmetry, PDNs offer some advantages in terms of robustness or fault tolerance over other networks of similar cost.

We begin by considering the effects of removing a single edge from a PDN. Does this action increase the shortest distance between any pair of nodes in the PDN? Unfortunately, it does. Note that whenever the shortest distance between a pair of nodes is 2, there are two edge-disjoint shortest paths so that the removal of a single edge never increases the length of the shortest path. When the two nodes are directly connected, however, removal of the edge connecting them may increase the length of the shortest path between these nodes and, thus, the network diameter, to 3. This is because the destination node must now be reached via one of its neighbors which could be two hops away from the source node. This increase may not be a serious problem. However, guaranteed good communication performance in the presence of a small number of failures is highly desirable. The failure of a single link is certainly such a scenario. Note that the root of the problem above is the inclusion of 0 in the PDS. This motivates us to introduce the following:

**Definition 4.** *0-free perfect difference set or network*—A 0-free PDS does not contain 0 as an element. Many 0-free PDSs can

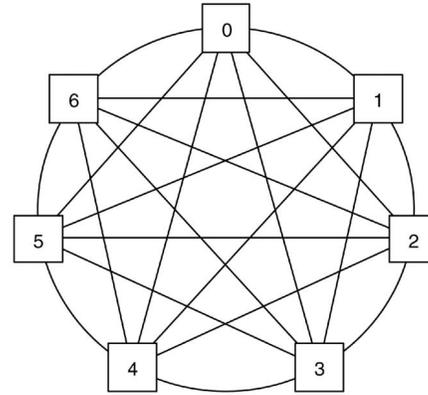


Fig. 5. PDN based on the PDS  $\{0, 1, 3\}$ , modified to correspond to the 0-free perfect difference set  $\{1, 2, 4\}$ .

be derived from a canonical PDS  $\{0, 1, s_2, \dots, s_\delta\}$ . For example, adding 1 to all the elements of a particular canonical PDS yields a corresponding canonical 0-free PDS. A perfect difference network based on a 0-free PDS is a 0-free PDN.

**Theorem 3.** *The diameter of a network derived by removing one link or node from a 0-free PDN remains 2.*

**Proof.** This property is a direct consequence of there being at least two edge and node-disjoint paths of length 2 between any two nodes, including those that are directly linked.  $\square$

Thus, for single node or link failures, a 0-free PDN provides a stronger fault tolerance guarantee than an ordinary PDN whose diameter may increase to 3. Note that a 0-free PDN that is based on a canonical 0-free PDS is still a chordal ring, with nodes appearing on the ring in numerical order. The 0-free version of the PDN based on the PDS  $\{0, 1, 3\}$  is shown in Fig. 5. For this example, the new version turns into the complete graph  $K_7$ , but generally, the node degree increases from  $2\delta$  to  $2\delta + 2$ , which is still  $O(n^{1/2})$ .

As a network grows in size, the probability of multiple failures may no longer be negligible. It is, therefore, important to study the effects of multiple node failures on the topological parameters and performance of a PDN. Tolerance to multiple edge failures is also required in practice, but we do not deal with this problem here except by noting that the effects of  $f$  edge failures are always less severe than those of  $f$  node failures. A measure of fault tolerance in the presence of multiple node failures is captured in the following definition:

**Definition 5.** *Fault diameter* [11]—The fault diameter of a network of connectivity  $c$  is the largest possible diameter of a network that would result after  $c - 1$  or fewer nodes fail (or are otherwise removed). The constraint of at most  $c - 1$  node failures is imposed to ensure that all healthy nodes of the network remain connected in the worst case despite the node failures.

As the number of faulty nodes increases, guaranteeing no change in the diameter of PDNs may be no longer possible. However, if paths of length 3 or 4 are acceptable, then robustness is significantly improved. Each node has

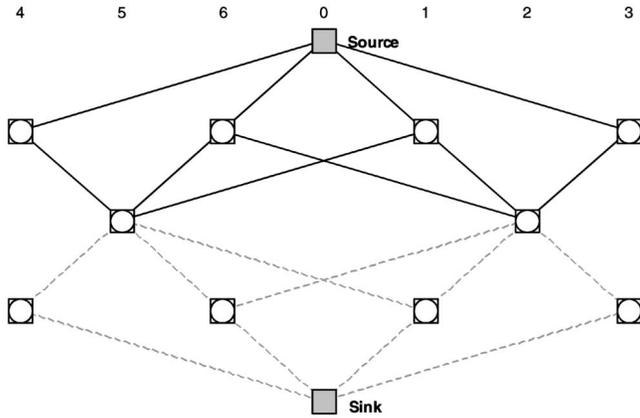


Fig. 6. Selected parts of an unfolded PDN, with links between nodes in adjacent rows, showing all paths of length 1 or 2 from source node 0 to every other node (solid lines) and from various nodes to sink node 0 (broken lines).

$2\delta$  neighbors, so there are potentially up to  $2\delta$  node-disjoint paths of length 3 from that node to a desired destination node. Even more paths of length 4 exist, which increases the likelihood of finding close to the maximum number of node or edge-disjoint paths of length 4 or less. Intuitively, this is seen by noting that the path can be established in two stages: from the source to an arbitrary intermediate node in 1-2 steps and from that node to the destination in 1-2 steps (Fig. 6). These ideas are formalized in the following:

**Theorem 4.** *The fault diameter of a PDN is no greater than 4.*

**Proof.** Given that the connectivity of a network is no greater than its minimum node degree, we prove the desired upper bound on fault diameter assuming  $2\delta - 1$  or fewer node faults. Consider arbitrary nodes  $x$  and  $y$  in a PDN with  $f$  faulty nodes. Let  $i$  neighbors of  $x$  and  $j$  neighbors of  $y$  be faulty, where  $i + j \leq f \leq 2\delta - 1$ . This means that  $x$  and  $y$  have  $2\delta - i$  and  $2\delta - j$  fault-free neighbors, respectively (Fig. 7). We can assume that these fault-free neighbors are distinct, for if a node belongs to both sets, there exists a path of length 2 between  $x$  and  $y$ , and nothing further needs to be proven. In the fault-free PDN, every one of the remaining  $\delta^2 - \delta$  nodes could be reached from  $x$  in two steps. With the  $i$  assumed faults among neighbors of  $x$ , at least  $\delta(2\delta - i)/2 - (f - i - j)$  distinct fault-free nodes within the oval-shaped region in Fig. 7 can be reached from  $x$  in two steps via its  $2\delta - i$  fault-free neighbors. This is because one of the two nodes  $x + s_a - s_b$  and  $x - s_a + s_b$  is

distinct from all others as we vary  $a$  and  $b$ . Similarly, at least  $\delta(2\delta - i)/2 - (f - i - j)$  distinct fault-free nodes in the middle region of Fig. 8 can be reached in two steps from  $y$ . If we show that the preceding two sets have a common member, then the conclusion that the fault diameter is no greater than 4 will be immediate. The latter is accomplished by contradiction. Assume that the two postulated sets of neighbors have no node in common. Then, the union of the two sets has

$$m = \delta(2\delta - i)/2 - (f - i - j) + \delta(2\delta - j)/2 - (f - i - j) = 2\delta^2 - (i + j)(\delta/2 - 2) - 2f$$

members. Based on the assumption  $f \leq 2\delta - 1$ , we can derive a lower bound for  $m$  by substituting the maximum possible value of  $2\delta - 1$  for  $f$  in the last expression, leading to

$$m \geq 2\delta^2 - (i + j)(\delta/2 - 2) - 2(2\delta - 1) = 2\delta^2 - 4\delta + 2 - (i + j)(\delta/2 - 2).$$

If  $\delta < 4$ , the last term just listed is positive and the contradiction  $m \geq \delta^2 - 3\delta - 1$  is obvious. For  $\delta \geq 4$ , we again substitute the maximum possible value of  $2\delta - 1$  for  $i + j$ , leading to  $m \geq 2\delta^2 - (2\delta - 1)(\delta/2 - 2) - 2(2\delta - 1) = \delta^2 + \delta/2$ . According to Fig. 7, this too is clearly a contradiction.  $\square$

It is known that, for  $D > 2$ , the fault diameter of a maximally resilient regular network whose connectivity is one less than its node degree is at least  $D + 1$ , that is, one more than its diameter. Because we have not shown PDNs to be maximally resilient, in the sense of their fault diameter being 3, and the condition  $D > 2$  is not satisfied by PDNs, known results about the fault diameter of highly resilient product networks [9] cannot be applied to multidimensional PDNs. It appears, however, that other combinatorial methods can be used to establish tight bounds on the fault diameter of such networks; this is being pursued by the authors.

One aspect of network robustness is its survivability, which requires lack of vulnerable spots along with ability to withstand limited failures or attacks [10]. Intuitively, lack of vulnerable spots is synonymous with "blandness" which means that careful study of the network will not reveal parts that are particularly attractive as targets of a malicious attack. An ability to withstand limited attacks, on the other hand, implies richness of connectivity so that disrupting certain nodes and/or links, via direct attacks or due to

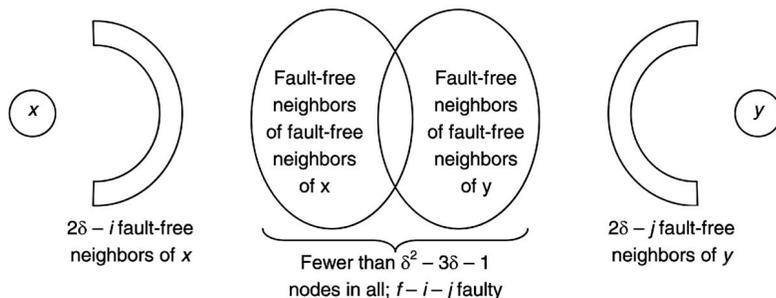


Fig. 7. Establishing the fault diameter of a PDN by examining the fault-free paths between arbitrary nodes  $x$  and  $y$ .

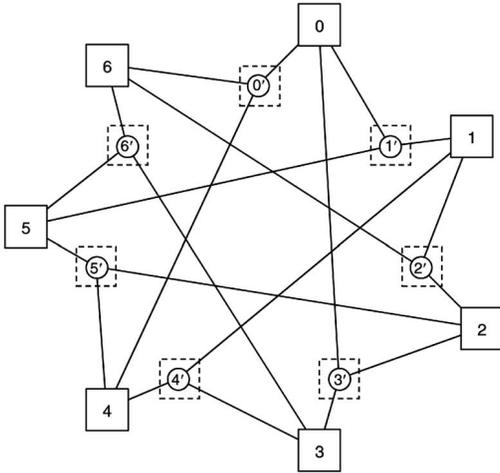


Fig. 8. Bipartite PDN, with  $n$  square host nodes and  $n$  circular switch nodes, viewed as a  $2n$ -node bipartite network (all nodes are hosts).

damage when nearby parts are destroyed, will not disconnect the network. PDNs are certainly both bland and richly connected.

Any PDN is obviously Hamiltonian. This is a desirable property for networks; in some cases, researchers go to great lengths to prove the Hamiltonicity of a newly proposed network or class of networks. In fact, usually many edge-disjoint Hamiltonian paths exist in a PDN, as stated in the following theorem:

**Theorem 5.** *For each  $s_i$  that is relatively prime with respect to  $n = \delta^2 + \delta + 1$ , there is a distinct edge-disjoint Hamiltonian path in the PDN. In particular, when  $n$  is prime with respect to each  $s_i$  in the PDS (e.g.,  $n$  is a prime number), the maximum possible number  $\delta$  of edge-disjoint Hamiltonian paths exist under the constraint of the node degree being  $2\delta$ .*

**Proof.** When  $s_i$  is relatively prime with respect to  $n$ , starting at any node  $x$  and traversing only  $s_i$  links in the forward direction will eventually take us through all nodes and back to  $x$ . Thus, for each  $s_i$  that is relatively prime to  $n$ , a separate edge-disjoint Hamiltonian path exists.  $\square$

An implication of Theorem 5 is that when  $k$  of the PDS members are relatively prime with respect to  $n$ , the associated PDN remains Hamiltonian in the presence of up to  $k - 1$  link failures. For example, the 133-node PDN based on the PDS  $\{0, 1, 3, 12, 20, 34, 38, 81, 88, 94, 104, 109\}$  remain Hamiltonian with up to 9 arbitrary link failures. Also, when several PDSs of the same order exist, the choice of using one or the other may be influenced by this property.

The bipartite PDN is also quite robust. Considering the network in Fig. 8, with uniform nodes (not nodes and switches), we note that the network diameter is 3. The host nodes replacing the original switch nodes are denoted by primed indices. Each such primed node is directly connected to several unprimed nodes and any pair of unprimed nodes are connected by a shortest path of length no greater than 2. As an example, node 0 in Fig. 8 is not connected to node 2' by any path of length less than 3, but there are several paths of the latter kind:  $0 \ 1' \ 1 \ 2'$ ,  $0 \ 3' \ 2 \ 2'$ ,  $0 \ 0' \ 6 \ 2'$ . These paths are node and edge-disjoint. In general, there would be  $\delta + 1$  such paths

through all switches connected to the source node, given that the interswitch diameter is also 2.

## 6 NETWORK COMPARISONS

In this section, we endeavor to compare PDNs with interconnection networks that have been studied in the past or that can be derived by combining a number of known networks. Even though some of the examples to follow represent architectures that are comparable to PDNs in terms of topological parameters and performance, it is still the case that PDNs provide additional useful design points. For example, the number of nodes in a product network of the form  $G \times G$  is always a perfect square. Thus, the number  $\delta^2 + \delta + 1$  of nodes in a PDN, which is roughly midway between  $\delta^2$  and  $(\delta + 1)^2$ , provides network sizes that would not be available in the aforementioned class of networks. Of course, the product of two networks of sizes  $\delta$  and  $\delta + 1$  supplies roughly the same number of nodes as a PDN. However, in the latter case, symmetry is sacrificed which may lead to uneven communication traffic and, ultimately, lower performance.

Alternative designs are valuable even if one cannot establish immediate advantages for them. Sometimes, the course of technological developments make certain alternatives more attractive than others. Examples include the surge in popularity of mesh and torus networks with the advent of very dense VLSI circuits, even though these architectures were at one time considered too inefficient, and the more recent attention to bus-based connection schemes with pipelined optical communication. Furthermore, the availability of multiple alternatives to a designer may allow a choice that better fits the application requirements.

**Example 1.** The product network  $K_\delta \times K_\delta$  is an instance of the generalized hypercube [6]. It has  $\delta^2$  nodes of degree  $d = 2\delta - 2$  and diameter  $D = 2$ . The average internode distance of  $K_\delta \times K_\delta$  is  $\Delta = [2\delta - 2 + 2(\delta^2 - 2\delta + 1)]/\delta^2 = 2(\delta - 1)/\delta$ , i.e., distance is 1 in the same row/column and 2 elsewhere. Bisection width is upper bounded by  $\delta^3/4$ . Routing is done as follows: If source and destination nodes are in the same row/column, then send in one step via the direct connection in the corresponding  $K_\delta$ . Otherwise, send in two step in one of two ways: first to the correct column, then to the correct row, or vice versa. There are two node and edge-disjoint shortest paths between any pair of nodes, except if they are in the same row/column. In the latter case, alternate paths of length 2 exist that can be used in the event of a link failure, the only exception being in the uninteresting case of  $\delta = 2$ . The preceding discussion shows that  $K_\delta \times K_\delta$  is quite comparable to PDNs in terms of topological parameters and performance.

**Example 2.** A bipartite version of the network defined in Example 1 is depicted in Fig. 9. The network has an interhost diameter of 4, host node degree of 2, and switch in and out-degree of  $\delta$ . This is conceptually similar to meshes with row/column buses, which have been studied extensively [8], [24], [26]. This similarity is highlighted in Fig. 9 in which switches are labeled as belonging to "rows" or "columns." The positive property of this architecture is that nodes are very simple. If switches are replaced by buses, a very low-cost network

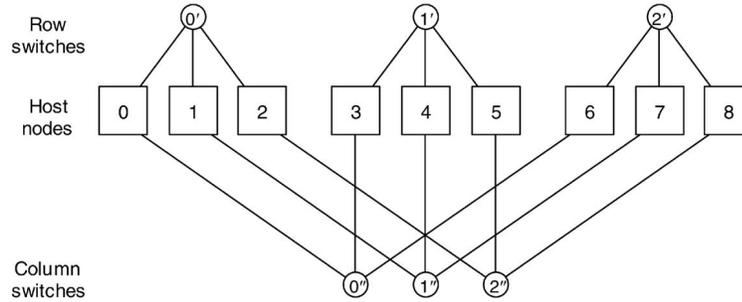


Fig. 9. Bipartite version of the product network  $K_3 \times K_3$  formed from hosts and switches.

results (albeit sacrificing performance, given that only one communication is allowed per bus in each step). With regard to topological and performance parameters, the network is quite similar to the bipartite form of PDN depicted in the example of Fig. 8. Comparing Fig. 9 to Fig. 8, we note that the node degree has been reduced from  $\delta + 1$  or  $\delta$  (depending on whether we compare to the next larger or smaller PDN) to 2 at the expense of increasing the interhost diameter from 2 to 4. This may be worthwhile, particularly if single-port communication forces us to use one of the outgoing links of each host per step, regardless of the host node degree.

**Example 3.** Swapped networks [16], [28], [29] provide another mechanism for building interconnection structures for comparison with PDNs. Consider a swapped network which consists of  $\delta$  copies of  $K_\delta$ , with node  $j$  of the  $i$ th copy connected to node  $i$  of the  $j$ th copy for every  $i$  and  $j$ , where  $i \neq j$ . This network has  $\delta^2$  nodes of degree  $\delta$ ; every node  $(i, j)$  has  $\delta - 1$  intracluster links in its  $K_\delta$  cluster  $i$  and one intercluster link to node  $i$  in  $K_\delta$  cluster  $j$ . The network diameter is 3 because node  $(i', j')$  is reachable from  $(i, j)$  in at most three steps:  $(i, j)$  to  $(i, i')$  in the source cluster, to  $(i', i)$  via an intercluster link, and finally to  $(i', j')$  in the destination cluster. The bisection width of this network is upper bounded by  $\delta^2/4$  (collapsing the clusters produces a  $K_\delta$  network) and is thus considerably smaller than that of a similar size PDN. Routing, broadcasting, or complete-exchange algorithms are quite simple for this network but under heavy traffic conditions, the communication performance is likely to be lower than that of a PDN due to the smaller bisection.

Table 2 provides a summary comparison between PDNs (both basic and bipartite versions) and the three example networks just discussed. Note that the picture provided by Table 2 is incomplete with regard to the two bipartite

networks in that the switches are ignored in determining the number of nodes and node degree.

Many other examples can be constructed for comparison with PDNs. We plan to perform more comparisons in the course of future research. However, it must be noted that general or application-independent comparison of interconnection networks is always difficult and inherently limited in usefulness [14]. Comparing topological parameters and performance in certain general communication problems is only a starting point. Unanticipated differences may be observed when specific applications, with their peculiar communication requirements, are dealt with. The point above further highlights the benefits of greater variety of choices for the designers of parallel and distributed computer systems.

## 7 CONCLUSION

Perfect difference networks constitute a class of robust, high-performance interconnection networks for parallel and distributed computation. Whereas PDNs may not be desirable for large networks with wired connectivity, they do offer attractive alternatives for wireless and optical interconnections and as smaller component networks in hierarchical architectures.

In this paper, we have studied the complexity and scalability aspects of PDNs and compared them to interconnection networks with similar cost/performance, including certain generalized hypercubes and their hierarchical variants. We also presented various routing algorithms for PDNs and demonstrated, via a general emulation result, that algorithms developed for a complete network can be emulated with optimal slowdown on a PDN of the same size. Although other interconnection networks with topological and performance parameters similar to PDNs exist, we

TABLE 2  
Summary of Some Network Parameters

Network $\rightarrow$	Ex. 1: $K_\delta \times K_\delta$	Ex. 2: Fig. 9	Ex. 3: $Swap(K_\delta)$	Basic PDN	Bipartite PDN
Nodes	$\delta^2$	$2\delta^2$	$\delta^2$	$\delta^2 + \delta + 1$	$\delta^2 + \delta + 1$
Degree	$2\delta - 2$	2	$\delta$	$2\delta$	$\delta + 1$
Links	$\delta^2(\delta - 1)/2$	$4\delta^2$	$\delta(\delta - 1)(\delta + 1)/2$	$\delta(\delta^2 + \delta + 1)$	$(\delta + 1)(\delta^2 + \delta + 1)$
Diameter	2	4	3	2	2
Avg. dist.	$2(\delta - 1)/\delta$	2	$\cong 3$	$2\delta/(\delta + 1)$	2
Bisection	$\leq \delta^3/4$	$\cong \delta^2$	$\leq \delta^2/4$	$\geq (\delta + 1)(\delta^2 + \delta + 1)/4$	?

In the case of Example 2 and bipartite PDN, only host nodes are counted.

still view PDNs as worthy additions to the repertoire of computer system designers. Alternative network topologies offer additional design points that can be exploited to accommodate the needs of new and emerging technologies. Further study is needed to resolve some open questions and to derive more accurate cost/performance comparisons for PDNs and their derivatives.

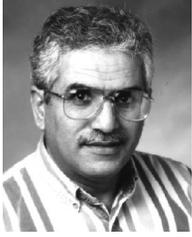
Although PDNs are interesting and important as asymptotically optimal diameter-2 interconnection networks, it is much more likely that hybrid or composite networks, involving PDNs as component structures, prove useful for practical applications. Here, we have dealt with multidimensional PDNs, and swapped networks based on PDNs, as specific examples of such networks, but many other hybrid or composite structures are possible.

In the area of robustness and fault tolerance, results thus far are encouraging. The rich connectivity of PDNs makes it possible to circumvent faulty nodes and links with little or no increase in routing distance and with negligible performance impact. In particular, we proved that PDNs remain Hamiltonian in the presence of many link faults and that their fault diameter is no greater than 4. PDNs offer many of the same robustness features of Cayley graphs that have recently become the focus of renewed attention [27]. Open problems for PDNs in this area include the determination of fault diameter for 0-free PDNs and other parameters quantifying the robustness of PDNs in general.

In a companion paper [19], we have noted possible extensions to the notion of perfect difference sets, for example, by allowing sums as well as differences to be used to reach other nodes in 2 or other suitably small number of steps. These variations have implications to the problems of routing and to robustness properties of the resulting networks. We plan to study these variations and the resulting interconnection networks in the immediate future.

## REFERENCES

- [1] S.B. Akers and B. Krishnamurthy, "A Group-Theoretic Model for Symmetric Interconnection Networks," *IEEE Trans. Computers*, vol. 38, no. 4, pp. 555-566, Apr. 1989.
- [2] A.-E. Al-Ayyoub and K. Day, "The Cross Product of Interconnection Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 8, no. 2, pp. 109-118, Feb. 1997.
- [3] A.-E. Al-Ayyoub and K. Day, "Comparative Study of Product Networks," *J. Parallel and Distributed Computing*, vol. 62, pp. 1-18, 2002.
- [4] B.W. Arden and H. Lee, "Analysis of Chordal Ring Networks," *IEEE Trans. Computers*, vol. 30, no. 4, pp. 291-295, Apr. 1981.
- [5] J.-C. Bermond, F. Comellas, and D.F. Du, "Distributed Loop Computer Networks: A Survey," *J. Parallel and Distributed Computing*, vol. 24, no. 1, pp. 2-10, Jan. 1995.
- [6] L.N. Bhuyan and D.P. Agrawal, "Generalized Hypercube and Hyperbus Structures for a Computer Network," *IEEE Trans. Computers*, vol. 33, no. 4, pp. 323-333, Apr. 1984.
- [7] G. Bilardi and F.P. Preparata, "Horizons of Parallel Computation," *J. Parallel and Distributed Computing*, vol. 27, pp. 172-182, June 1995.
- [8] D.A. Carlson, "Modified Mesh-Connected Parallel Computers," *IEEE Trans. Computers*, vol. 37, no. 10, pp. 1315-1321, Oct. 1988.
- [9] K. Day and A.-E. Al-Ayyoub, "Minimal Fault Diameter for Highly Resilient Product Networks," *IEEE Trans. Parallel and Distributed Systems*, vol. 11, no. 9, pp. 926-930, Sept. 2000.
- [10] A.M. Hobbs, "Network Survivability," *Applications of Discrete Math.*, J. G. Michaels and K. H. Rosen, eds., McGraw-Hill, pp. 332-353, 1991.
- [11] M.S. Krishnamoorthy and B. Krishnamurthy, "Fault Diameter of Interconnection Networks," *Computers & Math. with Applications*, vol. 13, nos. 5/6, pp. 577-582, 1987.
- [12] S. Lakshmirarahan, J.-S. Jwo, and S.K. Dahl, "Symmetry in Interconnection Networks Based on Cayley Graphs of Permutation Group: A Survey," *Parallel Computing*, vol. 19, pp. 361-401, 1993.
- [13] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, and Hypercubes*. Morgan Kaufmann, 1992.
- [14] K.J. Liszka, J.K. Antonio, and H.J. Siegel, "Problems with Comparing Interconnection Networks: Is an Alligator Better than an Armadillo?" *IEEE Concurrency*, vol. 5, no. 4, pp. 18-28, Oct.-Dec. 1997.
- [15] B. Parhami, *Introduction to Parallel Processing: Algorithms and Architectures*. Plenum Press, 1999.
- [16] B. Parhami, "Swapped Interconnection Networks: Topological, Performance, and Robustness Attributes," *J. Parallel and Distributed Computing*, to appear.
- [17] B. Parhami and D.-M. Kwai, "Challenges in Interconnection Network Design in the Era of Multiprocessor and Massively Parallel Microchips," *Proc. Int'l Conf. Comm. in Computing*, pp. 241-246, June 2000.
- [18] B. Parhami and D.-M. Kwai, "Incomplete k-ary n-cube and Its Derivatives," *J. Parallel and Distributed Computing*, vol. 64, no. 2, pp. 183-190, Feb. 2004.
- [19] B. Parhami and M. Rakov, "Perfect Difference Networks and Related Interconnection Structures for Parallel and Distributed Systems," *IEEE Trans. Parallel and Distributed Systems*, vol. 16, no. 8, pp. 714-724, Aug. 2005.
- [20] M. Rakov, "Method of Interconnecting Nodes and a Hyperstar Interconnection Structure," US Patent No. 5 734 580, Mar. 1998.
- [21] M. Rakov, "Multidimensional Hyperstar and Hyperhub Interconnection Methods and Structures," US Patent Application No. 09/410 175, Sept. 1999.
- [22] M. Rakov, "Hyperstar and Hyperhub Optical Networks Interconnection Methods and Structures," US Patent Application No. 09/634 129, Aug. 2000.
- [23] M. Rakov and J. Mackall, "Method of Interconnecting Functional Nodes and a Hyperstar Interconnection Structure," US Patent No. 6 330 706, Dec. 2001.
- [24] M.J. Serrano and B. Parhami, "Optimal Architectures and Algorithms for Mesh-Connected Computers with Separable Row/Column Buses," *IEEE Trans. Parallel and Distributed Systems*, vol. 4, no. 10, pp. 1073-1080, Oct. 1993.
- [25] J. Singer, "A Theorem in Finite Projective Geometry and Some Applications to Number Theory," *Trans. Am. Math. Soc.*, vol. 43, pp. 377-385, 1938.
- [26] L.D. Wittie, "Communication Structures for Large Networks of Microcomputers," *IEEE Trans. Computers*, vol. 30, no. 4, pp. 264-273, Apr. 1981.
- [27] W. Xiao and B. Parhami, "Some Mathematical Properties of Cayley Digraphs with Applications to Interconnection Network Design," *Int'l J. Computer Math.*, to appear.
- [28] C.-H. Yeh and B. Parhami, "Swapped Networks: Unifying the Architectures and Algorithms of a Wide Class of Hierarchical Parallel Processors," *Proc. Int'l Conf. Parallel and Distributed Systems*, pp. 230-237, June 1996.
- [29] C.-H. Yeh and B. Parhami, "Hierarchical Swapped Networks: Efficient Low-Degree Alternatives to Hypercube and Generalized Hypercube," *Proc. Int'l Symp. Parallel Architectures, Algorithms, and Networks*, pp. 90-96, June 1996.
- [30] S.G. Ziavras, Q. Wang, and P. Papathanasiou, "Viable Architectures for High-Performance Computing," *The Computer J.*, vol. 46, no. 1, pp. 36-54, 2003.



**Behrooz Parhami** received the PhD degree in computer science from the University of California, Los Angeles, in 1973. Presently, he is a professor in the Department of Electrical and Computer Engineering, University of California, Santa Barbara. His research deals with parallel architectures and algorithms, computer arithmetic, and reliable computing. In his previous position with Sharif University of Technology in Tehran, Iran (1974-1988), he was also involved

in the areas of educational planning, curriculum development, standardization efforts, technology transfer, and various editorial responsibilities, including a five-year term as editor of *Computer Report*, a Farsi-language computing periodical. Dr. Parhami's technical publications include more than 220 papers in journals and international conferences, a Farsi-language textbook, and an English/Farsi glossary of computing terms. Among his latest publications are two graduate-level textbooks on parallel processing (Plenum, 1999) and computer arithmetic (Oxford, 2000) and an introductory textbook on computer architecture (Oxford, 2005). Dr. Parhami is a fellow the IEEE and the IEEE Computer Society, a chartered fellow of the British Computer Society, a member of the Association for Computing Machinery, and a distinguished member of the Informatics Society of Iran for which he served as a founding member and president from 1979-1984. He also served as chairman of the IEEE Iran Section (1977-1986) and received the IEEE Centennial Medal in 1984.



**Mikhail A. Rakov** graduated (summa cum laude) from L'vov Polytechnic Institute in 1956 with a BSci degree in electrical engineering. He received the PhD degree from the Kiev Institute of Electrotechnics in 1962 and a DSci degree from the Moscow Institute of Energetics in 1971. From 1959 to 1993, he held positions of increasing responsibility in the Institute of Physics and Mechanics, Ukrainian Academy of Sciences, where he conducted research on

multiple-valued logic, abstract algebraic systems, digital signal processing, and network architecture. The main focus of this research was on developing practical applications for advanced mathematical constructs. He was appointed a professor of information theory and computer science in 1978 and was elected a member of the International Academy of Informatization in 1993. Dr. Rakov is author or coauthor of 10 scientific books, more than 250 published articles, and more than 100 patents in the former Soviet Union and the US. Since emigrating to the US, he has been working as an independent scientific consultant. He has been affiliated with the Computer Science Department at the University of California, Santa Barbara, as a professional researcher, beginning in 2002.

▷ **For more information on this or any other computing topic, please visit our Digital Library at [www.computer.org/publications/dlib](http://www.computer.org/publications/dlib).**