A family of network topologies with multiple loops and logarithmic diameter

Srabani Sen Gupta [a], Rajib K. Das [a],
Krishnendu Mukhopadhyaya [b], Bhabani P. Sinha [a,*]

[a] *Electronics Unit, Indian Statistical Institute, 203, B.T. Road, Calcutta 700 035, India*
[b] *Department of Mathematics, Jadavpur University, Calcutta 700 032, India*

Received 6 April 1995; revised 20 April 1996

Abstract

A new family of network topologies containing multiple loops is discussed in this paper. In the proposed structure, N processors are interconnected to form a graph $G(m, N)$, $m \geq 3$, where m is a parameter of the graph such that N is an even multiple of m and $(m-1) \times 2^{\lfloor (m-1)/2 \rfloor + 1} < N \leq m \times 2^{\lfloor m/2 \rfloor - 2}$. The graph $G(m, N)$ is hamiltonian with an average node degree $(3 + 1/m)$, when m is even and exactly 3 when m is odd. Whereas, the maximum node degree is 4. The diameter of $G(m, N)$ is upper bounded by $\lfloor 11m/8 \rfloor + 1$. A point to point routing algorithm has been presented. Implementation of ASCEND/DESCEND algorithms in $O(m)$ time has been discussed. It has been shown that in case of a single node failure, the diameter increases by at most 6.

Keywords: Ascend and Descend algorithms; Diameter; Network topology; Routing; Redundant binary representation

1. Introduction

Designing efficient schemes for interconnecting a large number of computing elements to form an integrated multiprocessor system has become an important area of recent research. Different network topologies exists in the literature [9,1] for this purpose. In designing a topology, low number of links per node (to reduce the cost of interconnection), small internode distances, large number of alternative paths between every pair of nodes (for improving the fault tolerance), etc. are some of the key

considerations. Mesh, ring, chordal ring, tree, hypercube, cube-connected cycle, etc. are a few among the popular interconnection topologies used for parallel and distributed systems.

The ring topology is widely used due to its structural symmetry and simplicity. The routing algorithm in a ring is very simple. But a ring with N nodes, has the diameter $\lfloor N/2 \rfloor$, leading to a large communication delay. The diameter can, however, be reduced if additional links are introduced within a ring. Some of the topologies resulting from such modifications are chordal ring [3], distributed loop network [4,6], etc., which have been widely studied in the literature. In a chordal ring, the degree of every node is 3, while the diameter is $O(\sqrt{N})$, N being the total number of nodes in the graph. In a distributed loop network the lower bound on the diameter is $(\sqrt{2N-1}-1)/2$, with the degree of every node increased to 4.

We propose here a new family of network topologies, by adding a few extra links over the ring. A topology in this family has nodes with degrees only 2, 3, and 4 (average node degree is less than or equal to 3.25) and has the diameter upper bounded by $\lfloor 11m/8 \rfloor + 1$, where m is a parameter of the graph such that $m \geqslant 3$, N is an even multiple of m and $(m-1) \times 2^{\lfloor (m-1)/2 \rfloor + 1} < N \leqslant m \times 2^{\lfloor m/2 \rfloor - 1}$. This shows that the diameter of the topology is $O(\log N)$. In respect of diameter, this topology is thus superior to both the chordal ring and distributed loop network. The total number of links used in this network is less than that in a distributed loop network and is no more than $\frac{1}{12}$th of that in a chordal ring. An algorithm for point to point routing has also been presented. The *Ascend* and *Descend* types of algorithms [12] can be very efficiently implemented on this topology. Moreover, the successive values of N, for which the proposed topology can be defined, are at an interval of $2m < 4 \log N$. That is, if N and N', are two successive values of the total number of nodes with $N' \geqslant N$, then $N' - N = 2m$. This may be contrasted with other fixed degree topologies having $O(\log N)$ diameter, e.g., Moebius graph [10], de Bruijn graph [8], cube-connected cycle [12], etc., for which the successive values of N are at much larger intervals. For all such graphs, N' is at least $2N$. The proposed network graph is hamiltonian and in case of a single node failure, the diameter may increase at most by 6.

2. Description of the topology

We describe the topology in terms of a graph $G(m, N)$, having the following characteristics:

(a) N is the total number of nodes in the graph. Let the nodes be numbered as $0, 1, \ldots, N-1$.

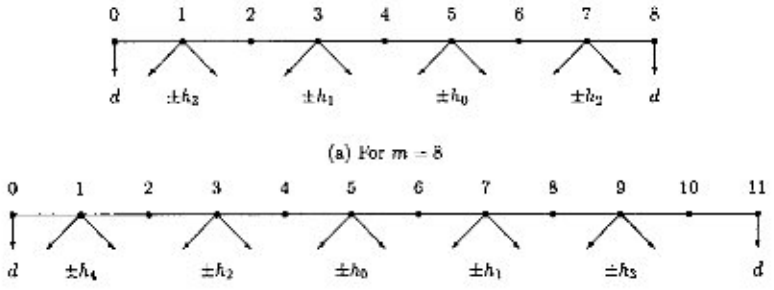
(b) m is a parameter of the graph such that $m \geqslant 3$.

(c) N is an even multiple of m such that $N = 2k \times m$, for some positive integer k.

(d) $(m-1) \times 2^{\lfloor (m-1)/2 \rfloor - 1} < N \leqslant m \times 2^{\lfloor m/2 \rfloor - 1}$.

(e) The nodes are connected by the following three types of edges (all operations below are treated under modulo N, unless otherwise mentioned):

(1) The node i is connected to the nodes $(i+1)$ and $(i-1)$. Thus the nodes are connected in the form of a cycle. We call these edges as c-edges (cyclic edges).



(a) For $m = 8$

(b) For $m = 11$

Fig. 1. Hop distribution in sector 0.

(2) For $0 \leqslant i \leqslant 2k-1$, the node $i.m$ is connected to the diametrically opposite node $(i.m + N/2)$. We call these edges as d-edges (diagonal edges).

After introducing d-edges, there are $2k$ number of nodes in the network, which are of degree 3. These degree 3 nodes divide the cycle, formed in (1), into $2k$ parts. We call each of these parts as a sector of length m. The sector j consists of the nodes, $\{j.m,\ j.m+1,\ j.m+2,\ldots,(j+1)m\}$, $0 \leqslant j \leqslant (2k-1)$.

(3) Nodes in different sectors are interconnected by a third type of edges, called as hops or h-edges, as described below.

Starting from the node $j.m+1$, in sector j, $0 \leqslant j \leqslant 2k-1$, h-edges are introduced at every alternate node (there are $\lfloor m/2 \rfloor$ such nodes in every sector). These connections are done by the following ways depending on the value of m. Let $\lfloor m/2 \rfloor - 1 = r$.

(i) The node $j.m+2i+1$ is connected to the nodes $[(j.m+2i+1) \pm m \times 2^{r-2i}]$, $i = 0, 1, 2, \ldots, \lfloor r/2 \rfloor$. These hops are denoted by $h_{\pm(r-2i)}$ respectively.

(iia) If r is even then the node $j.m + \lfloor m/2 \rfloor + 2i$ is connected to the nodes $[(j.m + \lfloor m/2 \rfloor + 2i) \pm m \times 2^{2i-1}]$, $i = 1, 2, \ldots, r/2$. These hops are denoted by $h_{\pm 2i}$ respectively.

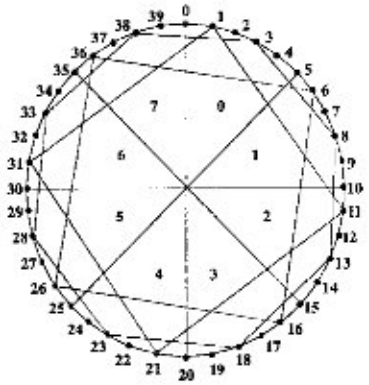


Fig. 2. The proposed graph $G(5, 40)$.

(iib) If r is odd then the node $j.m + \lfloor m/2 \rfloor + 2i + 1$ is connected to the nodes $[(j.m + \lfloor m/2 \rfloor + 2i + 1) \pm m \times 2^{2i}]$, by the hops $h_{\pm 2i}$, $i = 0, 1, \ldots, \lfloor r/2 \rfloor$.

An example of the hop distribution in sector 0 of the proposed topology when $m = 8$ and $m = 11$ is given in Fig. 1. Fig. 2 shows the complete connection pattern in $G(5, 40)$. It is to be noted here that the largest hop originated from a sector, is $h_{\lfloor m/2 \rfloor - 1}$. A hop h_i will be referred to as *even* (*odd*) if i is even (odd). The connection pattern shows that even and odd hops originate from different halves of a sector. The two hops connecting a vertex v to $v + m.2^i$ and $v - m.2^i$ will be referred to as $+h_i$ and $-h_i$ respectively.

Average node degree of this graph is $3 + 1/m$, when m is even and exactly 3 when m is odd. Thus the total number of edges in the graph is asymptotically $\sim 1.5\ N$.

Remark. If all the nodes in each sector (that is, the nodes jm, $jm + 1, \ldots, (j + 1)m - 1$ in sector j, $\forall j, 0 \leqslant j \leqslant N/m$) of $G(m, N)$ are coalesced to form a single node representing the whole sector then the resultant graph will be a supergraph of the circulant $G(N/m, \pm 1, \pm 2, \pm 2^2, \pm 2^3, \ldots, 2^{\lfloor \log(N/m) \rfloor - 1})$ [5,7] (or recursive circulant $G(N, d)$ for $d = 2$ [11]). Only in a special case, when $N = m.2^{\lfloor m/2 \rfloor - 1}$ the graph obtained by coalescing the nodes in each sector of $G(m, N)$ will be identical to $G(N/m, \pm 1, \pm 2, \pm 2^2, \pm 2^3, \ldots, 2^{\log(N/m) - 1})$. However, because of the typical degree distribution of the nodes in a sector, the extension of the results regarding diameter, routing, etc. of circulant graphs [5] or recursive circulant graphs [11] can not be directly extended to $G(m, N)$ and then need a separate treatment as given in the following sections. Moreover, the node degree (approximately equal to $\log_2 N$) in the recursive circulant graph increases with the total number of nodes N. In the proposed topology the average node degree is approximately equal to 3 and the maximum node degree is 4. In this regard the proposed topology is more cost effective.

3. Diameter of the network

In this section, we find an upper bound on the diameter D of the graph $G(m, N)$. To start with, we discuss about a restricted redundant binary number system to represent a node of the graph.

3.1. Restricted redundant binary representation

Definition. A *redundant binary* representation of a number $K = k_{r-1}k_{r-2} \cdots k_0$, is one in which each digit k_i, $0 \leqslant i \leqslant r - 1$, is an element of $\{0, 1, \bar{1}\}$ and $K = \sum_{i=0}^{r-1} 2^i k_i$.

Naturally, K does not have a unique representation in redundant binary system. For example, the binary number 0111011 has the equivalent representations $100\bar{1}10\bar{1}$ and $1000\bar{1}0\bar{1}$ in redundant binary.

Definition. A redundant binary representation, in which there is at least one zero bit in between two non-zero bits, will be termed as a *Restricted Redundant Binary* (RRB) representation.

Example 1. In the above example, $1000\bar{1}0\bar{1}$ is the RRB representation of the binary string 0111011.

Remark. This RRB representation is similar to the *canonical signed digit* (CSD) representation [2,13] of a positive integer for the radix 2. It follows from the results in [13] that the CSD (as well as RRB) representation of a number is unique and it can be computed sequentially by scanning the binary representation of the number from right to left. Thus, the RRB representation of a number N can be computed sequentially in $O(\log N)$ time.

Lemma 1. *In the RRB representation of N, the number of non-zero bits can be at most* $\lceil(\lceil \log N \rceil + 1)/2\rceil$.

Lemma 2. *In the RRB representation, the largest number M that can be obtained using b number of bits is given by*

$$M = \begin{cases} \frac{2}{3}(2^b - 1), & \text{if } b \text{ is even}, \\ \frac{2}{3}(2^b - 1) + \frac{1}{3}, & \text{otherwise}. \end{cases}$$

Proof. It is clear that for the largest number, every alternate bit starting from the most significant bit will be 1.

If b is even, then

$$L = 2^{b-1} + 2^{b-3} + \cdots + 2 = \tfrac{2}{3}(2^b - 1).$$

Otherwise,

$$L = 2^{b-1} + 2^{b-3} + \cdots + 2^0 = \tfrac{2}{3}(2^b - 1) + \tfrac{1}{3}. \qquad \square$$

Let w_x be the total number non-zero bits in the RRB representation of a number x. Further suppose that w_x^e and w_x^o be the number of non-zero bits corresponding to the even and odd powers ("0" is excluded) of 2 respectively, in the RRB representation of x.

Example 2. For $x = 100\bar{1}0\bar{1}01$, $w_x = 4$, $w_x^e = 2$ and $w_x^o = 1$.

It is clear that when x is odd, $w_x = w_x^e + w_x^o + 1$, otherwise $w_x = w_x^e + w_x^o$.

3.2. Upper bound on the diameter

To find the diameter it is enough to consider the paths from a node s in sector 0, to a node $d \leqslant N/2$, since all sectors look alike. Let d belong to the sector δ, $0 \leqslant \delta \leqslant \lfloor N/2m \rfloor$. Starting from s, we use hops to move across different sectors. But to avail a hop we may need to traverse some c-edges. With a view to minimizing the total walk along the c-edges we would take either odd hops or even hops (depending on the value of s and d) which emanate from only one half of every sector. h_0 may be included in both the cases. Since accessing h_l is equivalent to accessing h_{l-1} twice, $\lfloor m/2 \rfloor - 1 \geqslant l \geqslant 1$, it is always possible to get such a collection of hops. The method how we select the hops to reach sector δ starting from sector 0 is discussed below.

First we find the RRB representation of the sector difference δ (since the source is 0 here). The set of hops corresponding to the non-zero bits of this representation will be sufficient to cover these δ sectors. Without loss of generality, let us assume that $w_\delta^e \geqslant w_\delta^o$. Therefore, it is wise to convert odd hops into even hops. In that case, the required number of hops will be $w_\delta^e + 2w_\delta^o$ instead of $w_\delta^e + w_\delta^o$. Now, let us try to estimate a bound on this number.

Case 1: $0 \leqslant d \leqslant N/4$, i.e., $0 \leqslant \delta \leqslant \lfloor N/4m \rfloor$.

As $0 \leqslant \delta \leqslant \lfloor N/4m \rfloor$, the number of bits required for RRB representation of δ is $\lfloor m/2 \rfloor$.

For odd δ, the least significant bit (lsb) in RRB representation of δ is non-zero and hence the second lsb is 0. Among the remaining $\lfloor m/2 \rfloor - 2$ bits the number of non-zero bits will be at most $\lfloor (\lfloor m/2 \rfloor - 2)/2 \rfloor = \lfloor (m-2)/4 \rfloor$, by Lemma 1.

Thus, $w_\delta^e + w_\delta^o = \lfloor (m-2)/4 \rfloor$. After conversion, we need at most $2w_\delta^o + w_\delta^e + 1$ number of hops. Now, $2w_\delta^o + w_\delta^e + 1 \leqslant \lfloor \frac{3}{2}(w_\delta^o + w_\delta^e) \rfloor + 1 \leqslant \lfloor \frac{3}{2} \lfloor (m-2)/4 \rfloor \rfloor + 1$. Thus the maximum number of hops required to reach sector δ (using either even or odd hops along with some $\pm h_0$) is $\lfloor \frac{3}{2}(\lfloor (m-2)/4 \rfloor) \rfloor + 1$.

For even δ, $w_\delta^e + w_\delta^o = w_\delta \leqslant \lceil (\lfloor m/2 \rfloor - 1)/2 \rceil = \lfloor m/4 \rfloor$. Therefore, when we need any conversion, the number of hops required is at most $\lfloor \frac{3}{2}(\lfloor m/4 \rfloor) \rfloor$.

Hence, the maximum number of hops required to reach sector δ, under the restriction that either even or odd hops (h_0 may be included in both the cases) can be used, is

$$B_1 = \begin{cases} \left\lfloor \frac{3}{2}\left(\left\lfloor \frac{m}{4} \right\rfloor \right) \right\rfloor, & \text{when } m \bmod 8 \text{ is 0 or 1,} \\ \left\lfloor \frac{3}{2} \left\lfloor \frac{m-2}{4} \right\rfloor \right\rfloor + 1, & \text{otherwise.} \end{cases}$$

Case 2: $\lfloor N/4m \rfloor \leqslant \delta \leqslant \lfloor N/2m \rfloor$.

To reach this part of our network, we may utilize diagonal edges. But if we include a diagonal edge in our path, we would not use any $\pm h_{\lfloor m/2 \rfloor - 1}$ hop.

From sector 0, if we use a d-edge we will reach the boundary of the sectors $\lfloor N/2m \rfloor$. From there we have to cover a distance of j sectors (in backward direction) to reach sector δ, where $j = \lfloor N/2m \rfloor - \delta$. To traverse this distance we would not use hops $\pm h_{\lfloor m/2 \rfloor - 1}$. Here, by Lemma 2 we have, $0 \leqslant j \leqslant \frac{2}{3}(2^{\lfloor m/2 \rfloor - 1} - 1) + \frac{1}{3}$, which implies, $\lfloor N/2m \rfloor - \frac{1}{3}(2^{\lfloor m/2 \rfloor} - 1) \leqslant \delta \leqslant \lfloor N/2m \rfloor$. For such a δ, the maximum number of hops required to reach the sector δ can be enumerated as we have done in case 1. If B_2 represents this bound (including the diagonal edge), then

$$B_2 = \begin{cases} \left\lfloor \frac{3}{2}\left(\left\lfloor \frac{m-2}{4} \right\rfloor \right) \right\rfloor + 1, & \text{when } m \bmod 8 \text{ is 2 or 3,} \\ \left\lfloor \frac{3}{2}\left(\left\lfloor \frac{m}{4} \right\rfloor - 1 \right) \right\rfloor + 2, & \text{otherwise.} \end{cases}$$

Now, we are left with the case when $\lfloor N/4m \rfloor \leqslant \delta \leqslant \lfloor N/2m \rfloor - \frac{1}{3}(2^{\lfloor m/2 \rfloor} - 1) - 1$. This range is totally contained in the range $0 \leqslant \delta \leqslant \frac{2}{3}(2^{\lfloor m/2 \rfloor} - 1)$. Thus in this case, the

number of hops required to reach the destination sector δ is same as the bound obtained in case 1.

Hence for $0 \leqslant \delta \leqslant \lfloor N/2m \rfloor$, the number of hops required to reach sector δ, is at most $\max(B_1, B_2)$. It can be easily verified that $\max(B_1, B_2)$ is always B_2 $\forall m$.

So far we have identified the appropriate hops required to reach the destination sector and have also enumerated a bound on the number of hops. Now, we would find a specific order in which these hops are to be availed so that the walk along the cycle will be small enough to minimize the total path length. Let us introduce the following notations first.

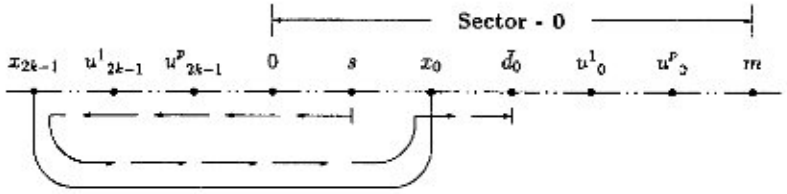
(1) x_j denotes the node in the sector j, from which hops of type 0 are originated. Thus x_j divides the sector j into two halves. Even and odd hops are emanated from these two halves.

(2) By $[a, b]$ we mean the set of the nodes $a, a+1, a+2, \ldots, b$.

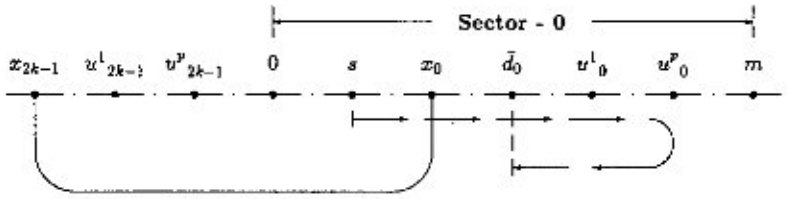
(3) Let H_δ be the set of hops of either even or odd types, required to reach sector δ. Suppose $|H_\delta| = p$. Let $u_j^1, u_j^2, \ldots, u_j^p$ be the nodes in sector j such that the hops in H_δ originate from these nodes and $u_j^1 < u_j^2 < \cdots < u_j^p$. Again, we call $[u_j^1, u_j^p]$ as the range of H_δ in sector j.

(4) Let the *projection* of d onto sector j be a node $\bar{d}_j$ in sector j, which is related to d by the equation $d = (\delta - j) \times m + \bar{d}_j$.

It is clear that to use the hops in H_δ (some of which may be repeated in the actual path), $\forall r$, $1 \leqslant r \leqslant p$, we must traverse through u_j^r at least once for some sector j, $0 \leqslant j \leqslant 2k-1$. Now, counting the number of c-edges involved in this traversal can be equivalently mapped to the problem of counting the number of c-edges required to reach $\bar{d}_0$ starting from s, with the restriction that any one of $\{u_0^r, u_{2k-1}^r\}$ is traversed at least once. We will now show that by appropriately choosing the order of the hops, this



(a) Traversal 1



(b) Traversal 2

Fig. 3. Traversal 1 and Traversal 2.

number can always be made less than or equal to m, by using *at most* one additional hop of type 0.

Without loss of generality let us assume that $s \leqslant \bar{d}_0$.

Case 1: s and $\bar{d}_0$ are in the different halves of the sector 0. Let us consider the following two subcases separately:

Subcase 1a: $[s, \bar{d}_0] \cap [u_0^1, u_0^p] = \emptyset$. In this case either $[u_0^1, u_0^p] \subseteq [\bar{d}_0 + 1, m]$ or $[u_0^1, u_0^p] \subseteq [0, s-1]$. Suppose, $[u_0^1, u_0^p] \subseteq [\bar{d}_0 + 1, m]$. Consider the situation in Fig. 3.

For $s + \bar{d}_0 < m$, use of the hops in H_δ requires the traversal from s to x_{2k-1} using c-edges, from x_{2k-1} to x_0 using an additional hop h_0 and then from x_0 to $\bar{d}_0$ using c-edges. Hence, the number of edges other than those in H_δ is $s + \lfloor m/2 \rfloor + 1 + \bar{d}_0 - \lfloor m/2 \rfloor = s + \bar{d}_0 + 1$. We call this traversal **traversal-1**. The maximum number of edges covered in this traversal is at most m.

For $s + \bar{d}_0 \geqslant m$, the required traversal is from s to u_0^p and from u_0^p to $\bar{d}_0$, using c-edges only, as shown in Fig. 3(b). Thus the number of edges other than those in H_δ, is

$$u_0^p \quad s + u_0^p - \bar{d}_0 \leqslant 2m \quad \left(s + \bar{d}_0\right) \leqslant m.$$

We call this traversal **traversal-2**.

For $[u_0^1, u_0^p] \subseteq [0, s-1]$, it can similarly be shown that the number of additional edges required is at most m.

Subcase 1b: $[s, \bar{d}_0] \cap [u_0^1, u_0^p] \neq \emptyset$. If $[u_0^1, u_0^p]$ is totally contained in $[s, \bar{d}_0]$, then the required path is direct from s to $\bar{d}_0$, using c-edges and the path length is $\bar{d}_0 - s$. Otherwise, it can be tackled in a similar way as we have done in subcase 1a. In both the situations the number of edges other than those in H_δ, will be at most m.

Case 2: s and $\bar{d}_0$ are in the same half of the sector. This case can also be treated in a similar fashion.

Remark. It is to be noted here that in case 2 a different approach will give us paths of smaller lengths. In this case we will use both even and odd types of hops corresponding to the RRB representation of δ, except those whose origins correspond to the points in $[s+1, \bar{d}_0 - 1]$. We will replace each of these hops in $[s+1, \bar{d}_0 - 1]$ by exactly two hops of smaller length. The details are given in [14]. The length of these paths are at most $\lfloor 5m/4 \rfloor + 2$.

Combining all the above results, the length $\hat{L}$, of the longest path between any two nodes is given by

$$\hat{L} = \begin{cases} \left\lfloor \frac{3}{2} \left(\left\lfloor \frac{m-2}{4} \right\rfloor \right) \right\rfloor + 1 + m, & \text{when } m \bmod 8 = 2 \text{ or } 3, \\ \left\lfloor \frac{3}{2} \left(\left\lfloor \frac{m}{4} \right\rfloor - 1 \right) \right\rfloor + 2 + m, & \text{otherwise.} \end{cases}$$

A little algebraic manipulation yields the following result.

Theorem 1. *The diameter D of the graph $G(m, N)$ is given by $D \leqslant \hat{L} \leqslant \lfloor 11m/8 \rfloor + 1$ if m mod 8 is 2, 4 or 5. Otherwise, $D \leqslant \lfloor 11m/8 \rfloor$.*

4. Routing algorithm

Input: (1) The source node s and the destination node d.
(2) The two parameters m and N of the topology.
Output: A path from s to d.

Procedure RRB(*sector-diff*, E, O)

/* E and O are two linear arrays of maximum size $\lfloor m/4 \rfloor$. These arrays are used to store even and odd bits of the RRB representation separately. "*sector-diff*" is the difference between the sector numbers of d and s. */

Step 1: Obtain the RRB representation of *sector-diff*.
Let it be $(b_L\ b_{L-1} \ldots b_2\ b_1\ b_0)$
Step 2: **for** $i = 1$ to $\lfloor m/4 \rfloor$ **do**
begin
$E[i] = b_{2i}$;
$O[i] = b_{2i-1}$;
end;
$b \leftarrow b_0$;

Example 3. For the source–destination pair (2, 42), the sector-diff is 5 and the values of E and O as computed by the procedure *RRB* are as follows:
$E[1] = 1$, $E[2] = 0$, $O[1] = 0$, $O[2] = 0$ and $b = 1$.

Procedure Hops (s, d, E, O, b, *diag-edge*)

/* Obtain hops in a specific order, in which they will be used to reach d, starting from s and store that order (along with the cyclic edges) in a final array. "*diag-edge*" is a boolean variable which indicates whether a diagonal edge is to be included in the path or not. The final array will be constructed on the basis of E, O, b and *diag-edge*. */

Step 1: Obtain $w^e_{sector\text{-}diff}$ and $w^o_{sector\text{-}diff}$.
Step 2: **if** $(w^e_{sector\text{-}diff} > w^o_{sector\text{-}diff})$ **then** convert all the odd hops to even hops and modify the elements of E and O in the way as described in Section 3.
else convert all the even hops to odd hops and modify the elements of E and O accordingly.
Step 3: /* To find the order in which the hops are to be used */
if $(\bar{s}_0 + \bar{d}_0 < m)$ **then** use the traversal-1
else use the traversal-2 (as discussed in Section 3)

Example 4. For the problem addressed in Example 3, the following values are computed by the procedure *Hops*: $w^e_5 = 1$, $w^0_5 = 0$. The order in which the edges will be accessed are as follows:

$$-c,\ -c,\ -c,\ h_2,\ -c,\ -c,\ h_0,\ h_0,\ -c,\ -c,\ -c$$

Here, $-c$ denotes a cyclic edge in counter-clockwise direction.

```
Procedure Find-Path (s, d, sector-diff)

/* This procedure finds the hops which are to be included in the path and the order H
in which they are to be used to reach d starting from s. */

Step 1: range ← ⌈⅔(2^⌊m/2⌋ − 1)⌉;
        if (0 < sector-diff ≤ range) then
        begin
          diag-edge ← "false";
          RRB(sector-diff, E, O);
        end;
        else
          begin
            diag-edge ← "true";
            /* to indicate whether a diagonal edge is to be included */
            sector-diff ← −(N/2m − 1 − sector-diff);
            RRB(sector-diff, E, O);
          end;
Step 2: s̄_0 ← s − ⌊s/m⌋ × m;
        d̄_0 ← d − ⌊d/m⌋ × m;
Step 3: /* assume that s̄_0 ≤ d̄_0 */
        Hops (s, d, E, O, b, diag-edge);

/* Main Program */

begin
  distance ← (d − s + N) mod N;
  sector-diff ← (⌊d/m⌋ − ⌊s/m⌋ + N/m) mod N/m
  if distance ≤ m then
  walk along the c-edges from s to d;
  else
    if sector-diff ≤ N/2m − 1 then
    Find-Path (s, d, sector-diff)
    else
      begin
        distance ← N − distance;
        if distance ≤ m then
        walk along the c-edges from s to d;
        else
          begin
            Find-Path (s, d, N/2m − sector-diff);
            Keeping the order same, change the sign of all the hops obtained;
          end;
      end;
end.
```

Example 5. According to the above algorithm, the following two paths are computed for the source destination pair (2, 42) and (13, 81) in $G(8, 256)$.

Path from 2 to 42: $2 \to 1 \to 0 \to 255 \to 31 \to 30 \to 29 \to 37 \to 45 \to 44 \to 43 \to 42$

Path from 13 to 81: $13 \to 21 \to 20 \to 19 \to 18 \to 17 \to 81$

It can be verified that execution of the above algorithm requires $O(\log N)$ steps. The path is computed once for all at the source node in terms of the hop types taken in order and attached to the transmitted message. As there are m different types of hops (including both positive and negative), $\log m$ bits are needed to designate each hop. Hence, the total number of bits required to specify a path is $O(m \log m) \approx O(\log N \log \log N)$.

5. Fault diameter

It is clear that the graph is biconnected. To find the diameter in presence of a single faulty node, we proceed as follows.

Let P be the path between two nodes s and d, obtained by the method discussed in section 3 with the path length $L(P) \leqslant \lfloor 11m/8 \rfloor + 1$. In presence of a single faulty node, if the faulty node f lies on the path P, then there may be two possible cases: (A) the faulty node lies on a sector other than the sector 0 and the destination sector, (B) the faulty node is in the destination sector. The case for the faulty node lying in sector 0 can be treated in the same way as case B.

Case A. Let h_q and h_r $(q < r)$ be two consecutive hops to be accessed in the path P and suppose while traversing along P, we enter the sector j by using the hop h_q and leave that sector by using the hop h_r. Let v_q and v_r be the respective nodes in sector j from where h_q and h_r originate.

The fault would affect the path if any one of the following occurs:

Case 1: The node v_r itself is faulty.

Case 2: The node v_r is a live node but the hop h_r would take us to such a node which is faulty.

Case 3: An intermediate node within the range $[v_q, v_r]$ is faulty.

We do not consider the case when the node v_q is faulty. Because in that case, the situation will be identical to the case 2 above for the sector from which we enter sector j. We consider below each of these cases separately.

Case 1. Instead of using a hop h_r, we can use four hops of type $(r-2)$. Thus, in this case the path length would be increased by three. Therefore, the path length would be at most $L(P) + 3 \leqslant \hat{L} + 3$.

Case 2. In this situation, we replace h_r by the hops $\{-h_r, -h_r, -h_r, h_{r-2}\}$ to bypass the fault. If h_r is used twice in P, then we replace those hops by $\{6(-h_r), 2h_{r+2}\}$. Thus the path length could be at most $L(P) + 6 \leqslant \hat{L} + 6$.

Case 3. We consider two subcases.

Subcase 3a: The faulty node is of degree 2. Let v_i and v_{i+2} be the two neighbors of the faulty node f, and $\pm h_i$ and $\pm h_{i+2}$ be the hops originated from these two nodes

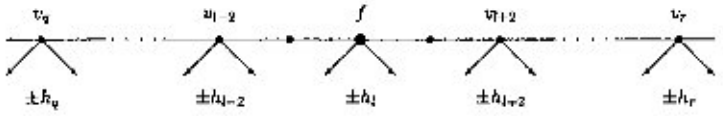


Fig. 4. Faulty node is of degree 4.

respectively. To bypass f, a sequence of five additional hops, namely, $\{-h_l, -h_l, -h_l, -h_l, h_{l-2}\}$ can be availed in between h_q and h_r. Thus the path length would be at most $L(P)+5 \leqslant \hat{L}+5$.

Subcase 3b: The faulty node is of degree 4. This situation is illustrated in Fig. 4. Assume, $q \neq l-2$ and $r \neq l+2$. To bypass f, a sequence of hops $\{-h_{l-2}, -h_{l-2}, -h_{l-2}, -h_{l-2}, -h_l, -h_l, -h_l, h_{l+2}\}$ can be used in between h_q and h_r. This will increase the path length by 8.

Now, $\hat{L}$ is a conservative estimate of the upper bound on $L(P)$. While computing $\hat{L}$, we assumed that the path P contains either all the even hops or all the odd hops at least once. But in this case, we see that there are at least three hops, namely, h_{l-2}, h_l and h_{l+2} which were not included in P. This reduces the upper bound on $L(P)$ to $\hat{L}-3$. Therefore, in presence of such a faulty node, the path length would be at most $L(P)+8 \leqslant \hat{L}+5$.

If $q=l-2$, then the sequence $\{h_q, h_r\}$ in P will be replaced by the sequence $\{-h_q, -h_q, -h_q, -h_l, -h_l, -h_l, h_{l-2}, h_r\}$. If $r=l+2$, then the sequence $\{h_q, h_r\}$ in P will be replaced by the sequence $\{h_q, h_{l-2}, h_{l-2}, h_{l-2}, h_{l-2}, h_l, h_l, h_l\}$. Thus, the path length will be increased by 6.

Case B. Let us now consider the case when the faulty node lies in the destination sector. Of course, we assume that $f \neq d$. Actually, the fault would affect the path P only if it lies in that part of the destination sector which is included in P. Let us call that part as the *restricted portion* of the destination sector. These restricted portions will be different for different traversals from s to $\bar{d}_0$ as discussed in the last part of Section 3. We shall consider those cases separately.

Case 1: When s and $\bar{d}_0$ are in the different halves of the sector 0.

Subcase 1a: $[s, \bar{d}_0] \cap [u_0^i, u_0^r] = \emptyset$.

(i) $s+d<m$. Here, in the destination sector (that is, in sector δ) the restricted portion is from x_δ to d.

If d is a degree-4 node, assume that the hops originating from d be of type q. To bypass the fault in the restricted portion, we shall use two additional hops $+h_q$ and $-h_q$ in P. While traversing along P, as soon as we encounter an origin of h_q (in some sector other than the destination sector) we will use that hop h_q. As a result, after using all the hops in H_δ, we will reach the sector $(\delta+2^q)$. From there we will avail $-h_q$ to reach the destination. Thus, in this case, we can bypass the fault at the expense of only two additional edges.

If d is not a degree-4 node, assume $\pm h_q$ originate from $d+1$. Here also the same technique of inserting $\{h_q, -h_q\}$ in P will work. But in this case four additional edges may be required (two cyclic edges along with two hops) to bypass the fault.

(ii) $s + d \geqslant m$. This situation can be tackled in a way similar to case (i) above.

Subcase 1b: $[u_0^1, u_0^p] \subseteq [s, \bar{d}_0]$. Here, corresponding to the path P, the restricted portion in the destination sector is from u_0^p to d. Let us consider the following two cases.

Case (a): If $\bar{d}_0$ and $[u_0^1, u_0^p]$ are in the different halves of the sector then to avoid the restricted portion in the destination sector, we will choose any one of the following two options:

Option 1: Without loss of generality, let us assume that the path P contains only even type of hops. Thus, the total number of hops included in P will be at most $2w_8^o + w_8^e + 1$. The new traversal can be done as follows. Starting from s, use only c-edges to reach x_0. From there use h_0 to reach x_1 in sector 1. From sector 1, start using hops in increasing order of magnitude. After using all the hops in H_8, use only c-edges to reach $\bar{d}$. Thus, in this case, the total path length will be

$$L(P_1) = \frac{m}{2} - s + 1 + \frac{m}{2} + m - \bar{d}_0 + 2w_8^o + w_8^e + 1.$$

Option 2: In this alternative path, we will use only odd type of hops. As a result, the range of H_8 will now be in the other half of the sector. That is, $[u_0^1, u_0^p]$ and $\bar{d}_0$ are now in the same half of the sector. But at this point $[u_0^1, u_0^p]$ may not be a subset of $[s, \bar{d}_0]$. The number of hops in this case will be $2w_8^e + w_8^o + 1$. Here, we shall bypass the fault by the same technique (at the expense of at most 4 edges) as we have taken in the first part of the subcase 1a. The total path length will be

$$L(P_2) = s + \frac{m}{2} + 1 + \bar{d}_0 - \frac{m}{2} + 2w_8^e + w_8^o + 1 + 4.$$

Now, $\min(L(P_1), L(P_2)) \leqslant (L(P_1) + L(P_2))/2 \leqslant 11m/8 + 5$. This shows that the path length may increase by at most 3, in presence of such a fault.

Case (b): If $\bar{d}_0$ and $[u_0^1, u_0^p]$ are in the same half of the sector then if we interchange s and d, the situation will be identical to the case (a) above.

Subcase 1c: $[u_0^1, u_0^p] - [s, \bar{d}_0] \neq \emptyset$.

This situation can be treated in a similar way as we have done in subcase 1a.

Case 2: s and $\bar{d}_0$ are in the same half of the sector.

With respect to the path P, discussed in the remark made in Section 3, the restricted region in the destination sector will be included in the portion from x_8 to d.

Let the hops that originate from either $\bar{d}_0$ or $\bar{d}_0 - 1$ be of type q according to the situation whether d is a node of degree 4 or not, respectively. Here, to bypass the fault, the sequence of hops (h_{q-1}, h_{q-1}, h_q) can be inserted in P.

From the above discussion, we can conclude that in presence of a single fault, the diameter of the topology can be increased at most by 6.

6. Implementation of ASCEND / DESCEND algorithms

When N is a power of 2, a class of parallel algorithms, known as ASCEND and DESCEND types of algorithms [12], can easily be implemented on the proposed network topology.

Suppose, $N = 2^q$ and the input data $a_0, a_1, \ldots, a_{N-1}$ are stored in the processors $P[0], P[1], \ldots, P[N-1]$, respectively. An algorithm is in the ASCEND class if a sequence of operations is carried out between a pair of data that are successively $2^0, 2^1, \ldots, 2^{q-1}$ processor locations apart. In DESCEND class of algorithms, the operations are carried out just in reverse order. These classes of algorithms have applications in the problems like *cyclic shift*, *bitonic merge*, *odd-even-merge*, *Fast Fourier Transform*, *shuffle*, *matrix transposition*, *bitonic sort*, etc. Now, We shall discuss the implementation of such algorithms in our case. For brevity, we shall discuss here only the ASCEND type of algorithms. For the ease of discussion, let us first renumber the nodes of $G(m, N)$.

Renumbering of nodes. Let $N = 2^q$ and $m = 2^r$. Therefore, $q = 2^{r-1} + r + 1$. Since there are $N/m = 2^{q-r}$ sectors, $G(m, N)$ contains 2^{q-r} cycles each of length $m + 1$, consists of m c-edges and a single hop h_0. Let us number these cycles as $0, 1, \ldots, 2^{q-r} - 1$, so that the cycles i and $(i + 1) \bmod 2^{q-r}$ have one node in common. The node in cycle 0 from which the hop h_0 is originated, is now renumbered as 0. The remaining nodes are numbered from 1 to $N - 1$ along the largest cycle of length N, so that the cycle i now consists of the renumbered nodes $\{im, im + 1, \ldots, (i + 1)m\}$. We would also represent any such renumbered node by an ordered pair (l, p), $0 \leqslant l \leqslant 2^{q-r} - 1$, $0 \leqslant p \leqslant m - 1 = 2^r - 1$, where l represents the cycle number which this node belongs to and p represents its distance from the node lm. Note that, since $p < m$, every node will have a unique representation by such an ordered pair. Thus, to address any of the N nodes, we require q bits in which the most significant $q - r$ bits would represent the cycle number and the least significant r bits would represent the position of the node in the corresponding cycle. Also, if a node is renumbered as n, then, $n = l.2^r + p$.

In our later discussions, we refer to a node by this renumbered value.

Before going to discuss the implementation, we now describe an ASCEND type of algorithm in the following two steps, where a basic operation between the two processors $P[i]$ and $P[r]$ has been indicated by OPER(i, j, $P[i]$, $P[r]$), when $r = i + 2^j$. $\text{bit}_j(i)$ denotes the jth least significant bit of the binary representation of i.

Proc ASCEND

Step 1: /* Process data elements within each cycle of length $m + 1$. */
  for each l, $0 \leqslant l \leqslant 2^{q-r} - 1$, **do in parallel**
    for $j = 0$ **to** $r - 1$ **do**
      for each p, $0 \leqslant p \leqslant 2^r - 1$ **do in parallel**
        if $\text{bit}_j(p) = 0$ **then**
          OPER(p, j, $P[(l, p)]$, $P[(l + 2^j, p)]$

Step 2: /* Process data elements across the cycles. */
  for $j = r$ **to** $q - 1$ **do**
  for each i, $0 \leqslant i \leqslant N - 1$ **do in parallel**
    if $\text{bit}_j(i) = 0$ **then**
      OPER(i, j, $P[i]$, $P[i + 2^j]$)

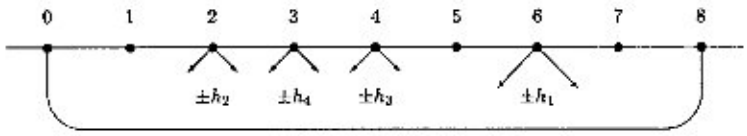


Fig. 5. Hop distribution.

Step 1 can be implemented in a similar way as it was discussed in [12]. This step can be executed on $G(m, N)$ in time linear in cycle length, that is, in $O(n)$ time. We shall now discuss the implementation of step 2 on $G(m, N)$. As an example, the hop distribution in cycle 0 of $G(2^3, 2^8)$ is illustrated in Fig. 5.

Initially, let us start with the assumption that the data element a_i is stored in the processor $P[i]$, $\forall i, 0 \leqslant i \leqslant N-1$. We shall now concentrate only on those operations which involve a_0, the data element initially stored in the processor $P[0]$. Step 2 of ASCEND algorithm demands that the operations between the pairs of data elements stored in the processors $(P[0], P[8])$, $(P[0], P[16])$, $(P[0], P[32])$, $(P[0], P[64])$, $(P[0], P[128])$ should be carried out successively. At first, using the hop h_0, the processor $P[0]$ and $P[8]$ can communicate with each other so that any operation can be carried out between a_0 and a_8. To perform the later operations, the data elements are to be shifted to some other nodes where the suitable hops are available. This can be made possible by successive shifting of each data element through all the nodes in the cycle, which it belongs to. From this point it is clear that the corresponding positions within a cycle must be same for the data elements among which operations are to be carried out. Now, by three successive shifts, the data element which is actually stored in $P[0]$ can be brought to the processor $P[9]$ from where the hop h_1 (which is actually a direct connection between cycle 0 and cycle 2) can be used to perform an operation with the element which was supposed to be stored in the processor $P[16]$ initially. Then to avail the next hop originating from the node 4, again two successive shifts are needed. The next available hop is h_3 which connects cycle 0 and cycle 8. With the help of this hop, an operation can be carried out between the elements which were initially stored in the processors $P[0]$ and $P[64]$. But, to fulfill the requirement of ASCEND algorithm the third operation must be between the data which were initially stored in the processors $P[0]$ and $P[32]$. Therefore, initially a_{32} must be stored in the processor $P[64]$. Similarly, a_{64} should be placed initially in $P[128]$ and a_{128} in $P[32]$. Thus, we see that some appropriate permutation must be specified for initial distribution of data elements among the processors. In general, the required permutation is described below in the inputting scheme.

Inputting of data elements. Let the sequence of hops originated from the nodes of the cycle l, $0 \leqslant l \leqslant 2^{q-r}-1$, in the order $(l, 1), (l, 2), \ldots, (l, m-1), (l+1, 0)$ be designated as $\langle h_{i_1}, h_{i_2}, \ldots, h_{i_\alpha} \rangle$, where $\alpha = 2^r - 1$. Clearly, $h_{i_\alpha} = h_0$. For example, the sequence of hops in any cycle of $G(2^3, 2^8)$ is $\langle h_2, h_4, h_3, h_1, h_0 \rangle$ (here, h_4 is the diagonal edge of length 2^7). The sequence of hops $\langle h_{i_1}, h_{i_2}, \ldots, h_{i_\alpha} \rangle$ can be alternatively designated by a sequence of numbers $\langle i_1, i_2, \ldots, i_\alpha \rangle$. Thus, for $G(2^3, 2^8)$, the sequence of hops is denoted by $\langle 2, 4, 3, 1, 0 \rangle$.

We now define a permutation π_m, associated with $G(m, N)$ as follows:

$$\pi_m = \begin{pmatrix} q-r & q-r-1 & \cdots & 0 \\ i_1 & i_2 & \cdots & i_\alpha \end{pmatrix}.$$

For $G(2^3, 2^8)$,

$$\pi_{2^3} = \begin{pmatrix} 4 & 3 & 2 & 1 & 0 \\ 2 & 4 & 3 & 1 & 0 \end{pmatrix}.$$

If we scan the top row of π_m from the right end, we get the figures correspond to the types of the hops to be successively taken for executing step 2 of the ASCEND algorithm. The bottom row of π_m, on the other hand indicates the order of the hop types actually existing in the network $G(m, N)$, when scanned from one end of a cycle of length m.

Let the processor $P[i]$ be placed at the node i. The N data elements $a_0, a_1, \ldots, a_{N-1}$ will be distributed among the processors $P[0], P[1], \ldots, P[N-1]$, in such a way that the element a_i will be stored in the processor $P[j]$, where, $i = (l, p)$, $j = (l', p)$ and the binary representation of l' is obtained by taking the permutation π_m on the binary representation of l.

Let us now discuss the implementation of step 2 on $G(m, N)$:

For a fixed j the computation corresponding to the *for* loop in step 2 can not be executed in one parallel step, because within a cycle only one node is actually connected to a node at 2^j distance apart, $\forall j$, $r \leqslant j \leqslant q-1$. Therefore, by means of successive circular shifts all the data elements in the cycle should be brought to that node, so that OPER(., j, ., .) can be executed. For each j, this step requires $2^r + 1$ $(= m+1)$ units of time. However, this computation can be pipelined and the total time required to execute step 2 can be reduced to only $O(m)$.

Step 2 can be explained in the following way.

Code section for the processor (l, p) in Step 2:

/* Assume that $\bar{l} = \pi_m^{-1}(l)$, where π_m^{-1} denotes the inverse permutation of π_m. That is, the binary representation of $\bar{l}$ is obtained by taking the permutation π_m^{-1} on the binary representation of l. Moreover, if (l, p) is a degree-4 node then assume that $\pm h_k$ originate from that point. Let b represent the kth bit of $\bar{l}$. The value of b can be either 0 or 1. */

for $i = 1$ to $2(m+1)$ **do**
begin
Step (2a):
if $((l, p)$ is a degree 4 node) **then**
  if $m - p + 2 \leqslant i \leqslant 2m - p + 1$ **then**
    if $(b = 0)$ **then**
    use $+h_k$ to communicate with the processor $(l + 2^k, p)$ to perform any operation on the data elements stored in these two processors;
    /* If $(p = 0)$ then perform this operation on the data which the processor $(l, 0)$ has obtained from $(l, 1)$. */

Table 1
Comparison of $G(m, N)$ with different topologies

No. of nodes (N)	Chordal ring		Distributed loop network		Proposed network	
	No. of links	Diameter	No. of links	Diameter	No. of links	Diameter
96	144	$\geqslant 9$	192	$\geqslant 7$	152	$\leqslant 8$
256	384	$\geqslant 15$	512	$\geqslant 11$	400	$\leqslant 11$
640	960	$\geqslant 25$	1280	$\geqslant 18$	992	$\leqslant 14$
1536	2304	$\geqslant 38$	3072	$\geqslant 28$	2368	$\leqslant 17$
3584	5376	$\geqslant 59$	7168	$\geqslant 42$	5504	$\leqslant 19$

else
use $-h_k$ to communicate with the processor $(l - 2^k, p)$ to perform any operation on the data elements stored in these two processors;
/* If $(p = 0)$ then perform this operation on the data which the processor $(l, 0)$ has obtained from $(l, 1)$. */
Step (2b):
if $(p \neq 0)$ **then**
send the data to $(l, p - 1)$
else
send the data which was obtained from $(l, 1)$ to $(l + 1, 0)$ and that which was obtained from $(l - 1, 0)$ to the processor $(l - 1, m - 1)$;**end**

Since the nodes $i.m(0 \leqslant i \leqslant 2^{q-r} - 1)$ are at the joint of two successive cycles, cycle i and $(i - 1)$ mod $2^{q-r} - 1$, care should be taken for proper pipeline of data elements within a cycle, by keeping two separate registers for the two cycles.

The *for* loop in step 2 is executed $2m + 2$ times. Moreover, for each i, $(\leqslant i \leqslant 2m)$, it takes two units of time, one for step 2a and another for step 2b. Therefore, the time required to execute step 2 is 4m + 2 units, that is O(log N) time. So, we can conclude that the ASCEND class of algorithms can be implemented on $G(m, N)$ in O(log N) units of time, when N is a power of 2.

7. Comparison with other topologies

Table 1 compares the number of links and diameter of $G(m, N)$ with those of the chordal ring and the distributed loop network, for different values of N.

8. Conclusion

A new family of graphs with constant node degree and low diameter has been introduced. With a slight modification in the definition, this family of graphs can be

constructed for all most all possible values of N. Given N, if N' is the nearest number to N, $N \leqslant N'$, on which the graph is defined, then construct a graph with N' nodes. Delete $N' - N$ number of degree 2 nodes, evenly from each sector. The diameter, $D(N)$, would be upper bounded by $D(N')$ and the maximum node degree would remain same as in the original graph.

References

[1] S.G. Akl, *The Design and Analysis of Parallel Algorithms* (Prentice-Hall, Englewood Cliffs, NJ, 1989).

[2] S. Arno and F.S. Wheeler, Signed digit representation of minimal Hamming weight, *IEEE Trans. Comput.* 42 (8) (1993) 1007–1010.

[3] B.W. Arden and H. Lee, Analysis of chordal ring network, *IEEE Trans. Comput.* 4 (1981) 291–295.

[4] J.C. Bermond and D. Tzvieli, Minimal diameter double-loop networks: Dense optimal families, *Networks* 1 (1991) 1–10.

[5] F.T. Boesch and J.F. Wang, Circulants and their connectivities, *J. Graph Theory* 8 (1984) 487–499.

[6] R.K. Das and B.P. Sinha, Optimal communication algorithms in distributed loop networks, *J. Parallel Distributed Computing* 30 (1995) 85–90.

[7] P.J. Davis, *Circulant Matrices* (John Wiley, New York, 1979).

[8] N.G. de Bruijn, A combinatorial problem, *Proc. Kon. Nederl. Acad. Wetensch.* A 49 (1946) 758–764.

[9] F.T. Leighton, *Introduction to Parallel Algorithms and Architectures: Arrays, Trees, Hypercubes* (Morgan Kaufmann, 1993).

[10] W.E. Leland and M.H. Solomon, Dense trivalent graphs for processor interconnection, *IEEE Trans. Comput.* 31 (1982) 219–222.

[11] J.H. Park and K.-Y. Chwa, Recursive circulant: A new topology for multicomputer networks, *ISPAN Proc.* (1994) 73–80.

[12] F.P. Preparata and J. Vuillemin, The cube-connected cycles: A versatile network for parallel computation, *Comm. ACM* (May 1981) 300–309.

[13] G.W. Reitweisner, Binary arithmetic, *Adv. in Comput.* 1 (1960) 231–308.

[14] S. Sen Gupta, R.K. Das, K. Mukhopadhyaya and B.P. Sinha, A multiple loop network with constant degree and logarithmic diameter, Tech. Rept. No. P & E/E/003-96, Indian Statistical Institute, Calcutta.