International Journal of Computer Science and Applications, ©2009 Technomathematics Research Foundation
Vol. 6, No. 1, pp. 26 – 39

# STUDY OF TOPOLOGICAL PROPERTY OF INTERCONNECTION NETWORKS AND ITS MAPPING TO SPARSE MATRIX MODEL

RAKESH KUMAR KATARE

*Department of Computer Science*
*A.P.S. University, Rewa (M.P.) India 486003*
*Katare_rakesh@yahoo.com*

N.S.CHAUDHARI

*Associate Professor*
*Nanyang Technological University, Singapore*
*asnarendra@ntu.edu.sg*

Sparse matrix elements on hypercube and perfect difference network have been studied. Gaussian symmetric factorization algorithm is being used to map the matrix on these two architectural topologies. A comparative study of hypercube and perfect difference network is done on the basis of topological properties. It has been shown that the access function or routing function contains topological properties. It has also been proved that the perfect difference network is not an instance of the generalized hypercube. A matrix multiplication algorithm using PDN has also been given in this paper. The complexity of these algorithm is $O(\log^3 n)$. Parallel algorithms for the symbolic factorization and triangular factor for sparse linear systems have also been demonstrated. Here we are using P-RAM model for the mapping of sparse matrix on interconnection networks. P-RAM model is a shared memory model and this model will survive as a theoretical convenient model of parallel computation and will used as a starting point for a methodology.

*Keywords*: Parallel algorithms, Sparse linear systems, Cholesky method, Hypercubes, Perfect difference network.

## 1. Introduction

This paper is primarily concerned with the topological property of interconnection network, especially for hypercube and perfect difference network.

Hypercubes are loosely coupled parallel processors based on the binary n-cube network; n-cube parallel processor consists of $2^n$ identical processors. In hypercube architecture the degree and diameter of the graph is same, because of this equality they achieve a good balance between the communication speed and the complexity of the topology network [15].

Perfect difference networks (PDNs) that are based on the mathematical notion of perfect difference sets have already been shown to comprise an asymptotically optimal method for connecting a number of nodes into a network with diameter 2 [10]. The hypercube architecture has many other limitations. Primarily, k-dimensional hypercube have $N=2^k$ vertices, so their structures are restricted to having exactly $2^k$ nodes. Because structures sizes must be a power of 2, there are large gaps in the sizes of systems that can be built with the hypercube. This restricts the number of possible nodes. The perfect

difference set establishes the structure that can be constructed for every prime power $n=p^r$. This provides a large advantage over the hypercube architecture, where structures exist only for the powers of 2 [11].

## 2. Background

We consider symmetric Gaussian elimination for the solution of the system of linear equations.

A x = b,

Where A is an n x n symmetric and positive definite matrix. Symmetric Gaussian elimination is equivalent to the square-root-free Cholesky method, i.e., first factoring A into the product $U^TDU$ and then forward and back solving to obtain x, and we often talk interchangeably about these two methods of solving A x = b.

A. Sherman[13] developed a row-oriented $U^TDU$ factorization algorithm for dense matrices A and consider modifications of it which take advantage of sparseness in A and U. We also discuss the type of information about A and U which is required to allow sparse symmetric factorization to be implemented efficiently. Here we will present a parallel algorithm for sparse symmetric matrix which will be mapped to hypercube and perfect difference network.

## 3. Analysis of Symmetric Gauss Elimination

In this section we suggest the actual testing operations that will be performed during Symmetric Gaussian elimination. Basically the Gaussian elimination is applied to transform matrices of linear equation to triangular form. This process may be performed in general by creating zeros in the first column, then the second and so on so forth. For k=1,2,...,n-1 we use the formulae

$$a_{ij}^{(k+1)} = a_{ij}^{(k)} - \left( \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \right) a_{kj}^{(k)}, \quad i, j > k \qquad \text{..... (1)}$$

and $$b_i^{(k+1)} = b_i^{(k)} - \left( \frac{a_{ik}^{(k)}}{a_{kk}^{(k)}} \right) b_k^{(k)}, \quad i > k \qquad \text{..... (2)}$$

Where $a_{ij}^{(1)} = a_{ij}$, i, j = 1, 2, .....n. The only assumption required is that the inequalities $a_{kk}^{(k)} \neq 0$, k=1,2, ....., n hold. There entries are called pivot in Gaussian elimination. It is convenient to use the notation.

$$A^{(k)} x = b^{(k)}$$

For the system obtained after (k-1) steps, k = 1, 2, ....., n with $A^{(1)} = A$ and $b^{(1)} = b$. The final matrix $A^{(n)}$ is upper triangular matrix [2]

As we know that in $a_{ij}^{(k+1)} = a_{ij}^{(k)} - \left(\frac{a_{ik}^{(k)}}{a_{ik}^{(k)}}\right) a_{jk}^{(k)}$ we are eliminating position $a_{ik}^{(k)}$

**Case 1 –** If $i \neq j$ then the above expression can be written as follows [7]

$$\left(a_{ij}^{(k)} - a_{ij}^{(k+1)}\right) = \left(\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\right) a_{jk}^{(k)} \qquad \text{..... (3)}$$

**Lemma 1:** if $(a_{ij}^{(k)} - a_{ij}^{(k+1)}) < 0$ then $a_{ik}^{(k)}$ or $a_{jk}^{(k)}$ is negative

4. Proof: -

$$(a_{ij}^{(k)} - a_{ij}^{(k+1)}) < 0$$

$$\Rightarrow \left(\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\right) a_{jk}^{(k)} < 0$$

$$\Rightarrow a_{ik}^{(k)} \quad or \quad a_{jk}^{(k)} \quad is \quad negative$$

$$\Rightarrow a_{ij}^{(k+1)} = a_{ij}^{(k)} + \left(\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\right) a_{jk}^{(k)}$$

**Lemma 2:** If $(a_{ij}^{(k)} - a_{ij}^{(k+1)}) > 0$ then $a_{jk}^{(k)}$ or $a_{ik}^{(k)}$ both negative or both positive and for pivot element $a_{ij}^{(k)}$ is always positive.

**Proof: -**

$$\left(a_{ij}^{(k)} - a_{ij}^{(k+1)}\right) > 0 \Rightarrow \left(\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\right) a_{jk}^{(k)} > 0$$

$$\Rightarrow a_{ij}^{(k)} > a_{ij}^{(k+1)} \quad then \quad a_{ij}^{(k+1)} \quad is \quad positive$$

$\Rightarrow a_{jk}^{(k)}$ or $a_{ik}^{(k)}$ both negative or both positive and for pivot element $a_{ij}^{(k)}$ is always positive.

**Lemma 3:** if $(a_{ij}^{(k)} - a_{ij}^{(k+1)}) = 0$ then no elimination will take place and one of the following condition will be satisfied.

(i) $a_{ik}^{(k)} = 0$ or (ii) $a_{jk}^{(k)}$=0, or both (i) & (ii) are zero.

5. Proof: -

if $(a_{ij}^{(k)} - a_{ij}^{(k+1)}) = 0$ then $a_{ij}^{(k)} = a_{ij}^{(k+1)}$ & $\left(\frac{a_{ij}^{(k)}}{a_{kk}^{(k)}}\right) a_{jk}^{(k)} = 0$

$\Rightarrow (a_{ik}^{(k)} . a_{jk}^{(k)}) = 0$

$\Rightarrow a_{ik}^{(k)}$=0 or $a_{jk}^{(k)}$=0 or both are zero.

$\Rightarrow$ $a_{ik}^{(k)}$ will remain same after elimination i.e. no elimination is required

**Case 2:** if $i = j$ then [7]

$$a_{ii}^{(k+1)} = a_{ii}^{(k)} - \left(\frac{a_{ik}^{(k)}}{a_{kk}^{(k)}}\right) a_{ik}^{(k)}$$

$$\Rightarrow a_{ii}^{(k+1)} = a_{ii}^{(k)} - \left(\frac{\left(a_{ik}^{(k)}\right)^2}{a_{kk}^{(k)}}\right)$$

**Lemma 4:** if $a_{ii}^{(k+1)} = 0$ then $a_{ik}^{(k)} = \sqrt{a_{ii}^{(k)} . a_{kk}^{(k)}}$

**Proof: -** If $a_{ii}^{(k+1)} = 0$ then $a_{ii}^{(k)} = \left(\frac{\left(a_{ik}^{(k)}\right)^2}{a_{kk}^{(k)}}\right) \Rightarrow a_{ii}^{(k)} . a_{kk}^{(k)} = \left(a_{ik}^{(k)}\right)^2$

$$\Rightarrow a_{ik}^{(k)} = \sqrt{a_{ii}^{(k)} . a_{kk}^{(k)}}$$

**Lemma 5:** if $a_{ik}^{(k)} = 0$ then $a_{ii}^{(k+1)} = a_{ii}^{(k)}$ no iteration will be take place.

**Proof: -** The result is obvious.

**Lemma 6:** if $\left(a_{ii}^{(k)} - a_{ii}^{(k+1)}\right) > 0$ then $a_{ik} > 0$

6. Proof: -

$$\left(a_{ii}^{(k)} - a_{ii}^{(k+1)}\right) > 0 \Rightarrow \left(\frac{\left(a_{ik}^{(k)}\right)^2}{a_{kk}^{(k)}}\right) > 0 \Rightarrow \left(a_{ik}^{(k)}\right)^2 > a_{kk}^{(k)}$$

$$\Rightarrow \left(a_{ik}^{(k)}\right)^2 > 0$$

**Lemma 7:** if $(a_{ii}^{(k)} - a_{ii}^{(k+1)}) = 0$ then $a_{ik} = 0$

7. Proof: -

$$(a_{ii}^{(k)} - a_{ii}^{(k+1)}) = 0 \Rightarrow a_{ii}^{(k)} = a_{ii}^{(k+1)} \Rightarrow \frac{\left(a_{ik}^{(k)}\right)^2}{a_{kk}^{(k)}} = 0$$

$$\Rightarrow \quad a_{ii}^{(k)} = 0$$

**Lemma 8:** if $(a_{ii}^{(k)} - a_{ii}^{(k+1)}) < 0$ then $a_{kk}^{(k)}$ will be eliminated.

8. Proof: -

$$(a_{ii}^{(k)} - a_{ii}^{(k+1)}) < 0 \Rightarrow a_{ii}^{(k)} > a_{ii}^{(k+1)} \Rightarrow a_{ii}^{(k+1)} = a_{ii}^{(k)} - \left(\frac{\left(a_{ik}^{(k)}\right)^2}{a_{kk}^{(k)}}\right)$$

In Gauss elimination, a nonzero position in position jk, implied that there was also a nonzero position in kj, which would cause fill to occur in position i, j, when row k is used to eliminate the element in position ik [3].

Here we present the following analysis when the matrix is symmetric and positive definite [7].

When i ≠ j in equation 3 and $\left(a_{ij}^{(k)} - a_{ij}^{(k+1)}\right)$ is less than 0 then the elements $a_{ik}^{(k)}$ or $a_{jk}^{k}$ is negative before elimination and when in equation 3 $\left(a_{ij}^{(k)} - a_{ij}^{(k+1)}\right)$ is greater than 0 then $a_{jk}^{(k)}$ or $a_{ik}^{(k)}$ both are negative or both are positive and for pivote element $a_{ij}^{(k)}$ is always positive and when in equation 3 $\left(a_{ij}^{(k)} - a_{ij}^{(k+1)}\right)$ is equal to zero then no elimination will take place and one of the following condition will be satisfied

(a) $a_{ik}^{(k)} = 0$ or (b) $a_{jk}^{(k)} = 0$ or both (a) and (b)

will be zero.

When i = j in the above explanation of equation 3 than for the first case the pivot element will be eliminated and for the second case the element which is to be eliminated will be greater than zero and for the third case elimination will not take place.

We suggested testing operation for variation occurring in the elements of Symmetric Gaussian elimination. There are more testing operations rather than arithmetic operations, so that the running time of the algorithm could be proportional to the amount of testing rather than amount of arithmetic operation, which will cause symbolic factorization to occur[13]. To avoid this problem Sherman pre computed "The set rak of

columns j ≥ k for which akj ≠ 0"; "The set ruk of columns j > k for which ukj ≠ 0";and "The set cuk of columns i < k for which uik ≠ 0";(where we assume that for each k, 1≤ k ≤ N) and showed that in implementation of this scheme, the total storage required is proportional to the number of non zeroes in A and U and that the total running time is proportional to the number of arithmetic operations on nonzeros. In addition to this the preprocessing required to compute these sets { rak} { ruk) and {cuk} can be performed in time proportional to the total storage.

## 4. Representation of Array Pattern of Processing Elements (P.Es.)

Consider a case of three dimensional array pattern with $n^3 = 2^{3q}$ (Processing Elements) PEs[2].

Conceptually these PEs may be regarded as arranged, in n × n × n array pattern. If we assume that the PEs are row major order, the PE (i,j,k) in position (i,j,k) of this array has index $in^2 + jn + k$ (note that array indices are in the range [0, (n-1)]. Hence, if $r_{3q-1}$ ..... $r_0$ is the binary representation of the PE position (i,j,k) then $i = r_{3q-1}$ ..... $r_{2q}$, $j = r_{2q-1}$ ..... $r_q$, $k = r_{q-1}$ ..... $r_0$ using *A(i,j,k), B(i,j,k)* and *C(i,j,k)* to represent memory locations in *P(i,j,k.*, We can describe the initial condition for matrix multiplication as

$$A(0,j,k) = A_{jk}$$

$$B(0,j,k) = B_{jk},\ 0 <= j < k,\ 0 <= k < n$$

$A_{jk}$ and $B_{jk}$ are the elements of the two matrices to be multiplied. The desired final configuration is

$$C(0,j,k) = C(j,k),\ 0 <= j < n,\ 0 <= k < n$$

Where

$$C_{jk} = \sum_{l=0}^{n-1} A_{jl} B_{lk} \qquad \text{.....} \qquad (4)$$

This algorithm computes the product matrix C by directly making use of (4). The algorithm has three distinct phases. In the first, element of A & B are distributed over the $n^3$ PEs so that we have $A(l,j,k) = A_{jl}$ *and* $B(l,j,k) = B_{lk}$. In the second phase the products $C(l,j,k) = A(l,j,k) * B(l,j,k) = A_{jl}B_{lk}$ are computed. Finally, in third phase the sum $\sum_{l=0}^{n-1} C(l, j, k)$ are computed.

The details are spelled out in Dekel, Nassimi and Sahni 1981[2]. In this procedure all PE references are by PE index (Recall that the index of PE(i,j,k) as $in^2 + jn + k$). The key to the algorithm of Dekel, Nassmi & Sahni in the data routing strategy $5q = 5 \log n$ routing steps are sufficient to broadcast the initial value through the processor array and to find the results.

**$O(\theta_A)$ Symbolic Factorization PRAM-CREW Algorithm**

Begin (1)

Repeat log n times do

For all (ordered) pair (i, j, k) 0 , $k \leq n$, $0 < i = j = k \leq n$ and q = log n in parallel do (2)

$a(2^{2q}i + 2^q j + k) = a(k, j)$

$u(2^{2q}i + 2^q j + k) = u(k, j)$

End for (2)

For all (ordered) pair (i, j, k), $0< k \leq n$, $\{j > k : u_{k,j} \neq 0\}$ and q=log n in parallel do (3)

$u (k, j) \leftarrow 0$

End for (3)

For all (ordered) pair (i. j. k), $0 < k < n$, and q = log n in parallel do (4)

For all (ordered) pair (i, j, k), $0< k < n$, and (5)

$j \in \{n \in a(k, j): n > k\}$ do

$u(k, j) \leftarrow u(k, j) \cup \{i\}$

end for (5)

For all (ordered) pair (i. J. k), $0 <k\leq n$ ,$i \in cu_k$ and q=log n in parallel do (6)

For all (ordered) pair (i, j, k) $0 <k\leq n$ , $j \in ru^k_i$ {= $\{j \in ru_i : j> k\}$}, and q=log n in parallel do (7)

if $j \notin u(k, j)$ then

$u(k, j) \leftarrow u(k, j) \cup \{j\}$

End if

End for (7)

End for (6)

End for (4)

End (1)

**Lemma 9:**The routing or access function $in^2+jn+k$ contains topological properties[4,5,6].

**Proof:** This access function is a polynomial of 3 dimensional discrete space. Where i, j, k are 3 dimensions and n is fixed. It gives a relationship of processing elements (i.e. there are $2^3$ connections (time)) that meet at each vertex of a hypercube means that the algorithm can be evaluated in polylogarithmic time using a polynominal ($in^2+jn+k$) of three dimensional discrete space. We can easily construct hypercube successively from lower dimensional cubes by using polynomial $in^2+jn+k$. A discrete space is a topological space in which all sets are isolated. We conclude that the access function by which we are mapping matrix elements will be pairwise continuous. It is shown in the implementation that because of the hamming distance between the processes the hypercube is modeled as a discrete space with discrete time. This access function is also used to map a matrix of three dimensions into RAM sequentially.

**Lemma 10:**Cost of storage and cost of retrieval of a matrix are proportional to each other in polylogarithmic parallel time using P-RAM with a polynomial number of processor.

**Proof:** For storage and retrieval of a matrix we use parallel iteration. Parallel iteration has the property of convergence in log n in parallel. It converges geometrically or at the rate of geometric progression therefore they are proportional to each other for a single value. From the above fact we can write that cost of retrieval is proportional to cost of storage

$\Rightarrow$ cost of retrieval = k x cost of storage, (Where k is a constant)

$\Rightarrow$ if $k \geq 1$ then it is a Dense matrix and if $k < 1$ then it is a sparse matrix.

## 5. Topological property of Hypercubes

In this section we adopt a graph-theoretic point of view and review the topological properties of hypercubes that makes them attractive. The basic reference for this section is Saad and Schultz (1988)[12].

A hypercube is a multidimensional mesh of nodes with exactly two nodes in each dimension. A d-dimensional hypercube consists of K nodes, where $K=2^n$.

1. A hypercube has n special dimensions, where n can be any positive integer (including zero).
2. A hypercube has $2^n$ vertices.
3. There are n connections (lines) that meet at each vertex of a hypercube.
4. All connections at a hypercube vertex meet at right angles with respect to each other [1], [14].
5. The Hypercube can be constructed recursively from lower dimensional cubes.
6. An architecture where the degree and diameter of the graph is the same then they will achieve a good balance between, the communication speed and the complexity of the topology network. Hypercube achieve this equality, which explains why they are one of the today's most popular design (e.g. i psc of intel corp., T-series of FPS, n-cube, connection machine of thinking machines corp.). [15]

## 6. Topological properties of Perfect Difference Network [8,9]

In this section we adopt the topological properties of PDN. The basic reference for this section is Behrooz Parhami and Mikhail Rakov [8, 9, 11].The 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.

**Definition 1**: Perfect Difference set (PDS)- A set $\{s_0,s_1,\ldots\ldots\ldots s_\delta\}$ of $\delta+1$ integers having the property that their $\delta^2+\delta$ differences $\delta_i - \delta_j$, $0 \leq i \neq j \leq \delta$ are congruent, modulo $\delta^2+\delta+1$, to the integers 1, 2,….., $\delta^2+\delta$ 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 $1<s_2<s_3$, $<\ldots\ldots<s_\delta$, Henceforth, we deal only with normal form PDSs of the form $\{0,1,s_2,\ldots\ldots,s_\delta\}$.

**Definition 2:** Perfect difference network (PDN) based on the PDS $\{0,1,s_2,\ldots\ldots s_\delta\}$ - There are $n=\delta^2+\delta+1$ nodes, numbered 0 to n-1, Node i is connected to nodes $i \pm 1$ and $i \pm s_j$ (mod n), for $2 \leq j \leq \delta$. Because all index expressions in this paper are evaluated modulo n, we will delete the qualifiers "mod n " throughout.

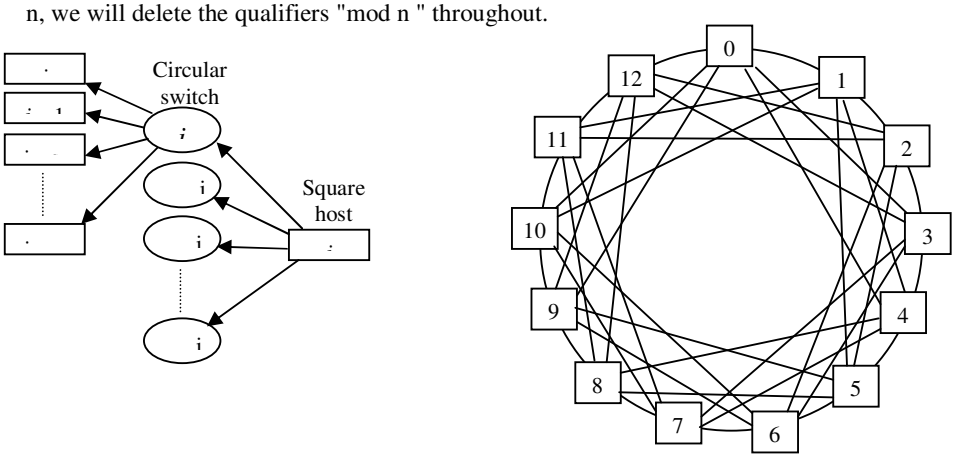


Fig 1.1: The chordal ring structure of PDN with n = 13 nodes based on the perfect difference set {0, 1, 3, 9} and the strategy for deriving a 2n - node bipartite PDN (with n circular and n square nodes) from an n-node PDN (a) Basic PDN with n = 13 nodes. (b) Deriving a bipartite PDN(Source[8, 9]).

The (basic) PDN just defined is an undirected chordal ring of degree, d= $2\delta$ and diameter D = 2 (Fig 1.1 b). 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 1.1 a].

Apart from the diameter in this section we will summarize two more topological properties of the basic PDN. The first one is average internode distance and the second one is bisection width[8,9].

1. Average internode distance: Each node has distance of 0 to itself, 1 to its 2δ neighbors, and 2 to the other $\delta^2$- δ nodes. Hence, $\Delta = [2\ \delta + 2(\delta^2 - \delta)]/n = 2\delta^2/n$. If we did not count the distance of a node to itself, the average internode distance would become $2\delta^2/(n - 1) = 2\delta /(\delta + 1)$. Hence the average inter node distance of a PDN of order δ is Δ = 252/n.
2. The upper bound of the PDN is $\min(2S'_{all}, nM_{odd} - S_{odd} + S_{even})$, where $M_{odd}$ is the number of odd elements in the PDS, $S_{even}$ and $S_{odd}$ represent the sum of all PDS elements that are even and odd, respectively, and $S_{all}$ is the sum of all $s'_i$ values for the PDS. For an element $s_i$ of a specific PDS of order δ, define $s'_i$ as $s_i$ if $s_i < n/2$ and as $n-s_i$ if $s_i > n/2$.
3. The total number of links going between odd and even nodes is

$$\sum_{\text{odd skips}} (n-s_j) + \sum_{\text{even skips}} s_j = nM_{odd} - S_{odd} + S_{even}$$

4.
5. The lower bound on the bisection width of PDN is [(δ + 1)(n + 1)/4]
6. Property 2 & 4 implies that the bisection width of PDN is Θ(n1.5), which is intermediate between the Θ (n) bisection of the hypercube and the Θ (n2) bisection of Kn (Complete Graph).
7. The calculation of bisection width for an arbitrary graph is an NP - complete problem.

## 7. A Comparative Study of Hypercube & Perfect Difference Network (PDN) [4].

1. A k-dimensional hypercube have $N=2^n$ vertices, so these structures are restricted to having exactly $2^n$ nodes. Because structure sizes must be a power of 2, there are large gaps in the sizes of systems that can be built with the hypercube. This severely restricts the number of possible nodes. A hypercube architecture has a delay time approximately equal to *2 log n* and has a "Skew", i.e. different delay times for different Interconnection nodes.

   An Intercomection structure includes a plurality of nodes and a plurality of information elements. The plurality of nodes are arranged in a regular array having a pre selected number N [11].

2. The cross product (Cartesian product or. simply product) of the q graphs, $G_i = (V_i, E_i)$, $0 \le i \le q - 1$, denoted as $G_{q-1} \times G_{q-2} \times ... \times G_0$, is a graph with $n_{q-1} \times n_{q-2} \times ... \times n_0$ nodes, each labeled with a distinct q-digit mixed-radix integer $x_{q-1}x_{q-2} ... x_0$ in the range 0 to $n_{q-1} \times n_{q-2} \times ... \times n_0 - 1$, so that nodes x and y are connected, if their labels differ only in one digit, say $x_j \neq y_j$, and $x_j$ is connected to $y_j$ in $G_j$ [8, 9]. The node degree of a product graph is the sum $d_{q-1} + d_{q-2} + ... + d_0$ of the node degrees for the component graphs and its diameter is the sum $D_{q-1} + D_{q-2} + ... + D_0$ of the diameters. A wide variety of networks can be built through cross-product composition. For instance, the q-cube (q-dimensional binary hypercube) is $K_2 \times K_2 \times ... \times K_2$. As a further example, the product of two $n^{1/2}$-node complete graphs is a network of node degree $2n^{1/2} - 2$ and diameter 2. More generally, $K_m \times K_m \times ... \times K_m$ is referred to as q-dimensional radix-m generalized hypercube. It has $n = m^q$ nodes, with node degree $d = q(m - 1)$ and diameter D = q.

3. A complete hyperstar interconnection structure comprising: a preselected number N of information elements; and a plurality of nodes arranged in a regular array having the preselected number N of columns and having two rows, each node (0,j) of the first row being coupled to nodes of the second row $(1,(j+s_i)(\text{mod } N))$, for j=0, 1, . . . ,N and i=0, 1, . . . , n, where $d_i$ are components of a Perfect Difference Set of order n, and the preselected number N of columns equals $n^2$ =n+1. Table-I shows the comparative study of basic parameter of Hypercube and perfect difference network

### 7.1. *Table - I*

Basic parameter of Hypercube and perfect difference network

| Graph | Links | Vertices/node | Degree | Diameter |
|---|---|---|---|---|
| n-cube | n | $2^n$ | n | n |
| PDN | $\delta(\delta^2+\delta+1)$ | $\delta^2+\delta+1$ | $2\delta$ | 2 |

Because of the study of topological properties of hypercube and PDN we claim the following result.

**Lemma 11:** The perfect difference network is not an instance of the generalized hypercube[4,5,6].

**Proof:** Perfect difference network (PDN) is based on the perfect difference set $\{0, 1, s_2, \ldots\ldots\ldots, s_\delta\}$. There are $n = \delta^2 + \delta + 1$ nodes, numbered 0 to n -1. Node i is connected via directed links to nodes $i \pm 1$ and $i \neq s_j \pmod n$, for $1 \leq j \leq \delta$. The preceding connectivity leads to a chordal ring of in and out degree $d = 2\delta$ and diameter D = 2. Because, for each link from node i to node j, the reverse link from node j to node i also exists, the network can be drawn as an undirected graph.

Where as hypercube is an undirected graph consisting of $n = 2^k$ vertices; if and only if the binary representation of their labels differ by one and only one bit. The most important property is that the degree of the graph and the diameter are always equal, which will achieve a good balance between the communication speed and the complexity of the topology network; these structures are restricted to having exactly $2^k$ nodes. Because structure sizes must be a power of 2, there are large gaps in the sizes of the system that can be built with the hypercubes. This severely restricts the number of possible nodes. A hypercube architecture has a delay time approximately equal to 2 log n and has a "skew", i.e. different delay times for different inter connecting nodes [11].

The properties of the hypercube and the perfect difference network can not be compared because of the property $2^d \neq \delta^2+\delta+1$.

Here we represent n matrix multiplication algorithm by using concept of PDN

**Line** **Procedure matmul**

begin matmul

1. for i = 1 to n do
2. for j = 1 to n do
3. A (i, j) ← $A(i-1, j+s_i(\text{mod } n))$ $(i \geq l)$
4. B (i, j) ← $B(i+1, j+s_i(\text{mod } n))$ $(i \geq l)$
5. end

```
6.  end
5.   for i = 1 to n do
8.        for j = 1 to n do
9.              A (i, j) ← A(i-1, j+s_i(mod n))
10.             B (i, j) ← B(i+1, j+s_i(mod n))
11.             C (i, j) = C(i, j) + A(i, j) * B(i, j)
12.       end
13. end
    end matmul.
```

If we know the matrix multiplication algorithm then we can easily explore this algorithm for sparsity of linear system. Here we are presenting algorithms for $U^TDU$ factorization and symbolic factorization of matrix using PDN.

***O*($\theta_A$) symbolic factorization PRAM-CREW Algorithm**

Begin (1)

Repeat *log n* times do

*For all (ordered) pair (i, j, k), 0< k ≤ n, {j > k : $u_{k,j}$ ≠ 0} and q=log n in parallel do (2)*

*u (k, j) ← 0*

*End for(2)*

*F*or all (ordered) pair *(i. j. k), 0 < k < n, and q = log n*

*in parallel do(3)*

*For all (ordered) pair (i, j, k), 0< k < n,(4)*

*j ∈ {n ∈ a(k, j): n > k} do(4)*

*u(k, j) ← u(k, j) ∪ {i}*

*end for(4)*

For all (ordered) pair *(i. J. k), 0 <k≤ n, i ∈ $cu_k$ and q=log n in parallel do(5)*

*F*or all (ordered) pair *(i. j. k) 0 <k≤ n ,j ∈ $ru_i$ and q=log n in parallel do(6)*

*if j ∉ u(k, j) then*

*u(k, j) ← u(k, j) ∪ {i}*

*End if*

*End for(6)*

*End for(5)*

*End for(3)*

*End.(1)*

## 8. Conclusion:

In mapping of data into the hypercube it was indicated that the data is mapped to its all possible neighbour processors in the n-cube which has hamming distance exactly by one bit, which makes like a tree structure of having leaf of all its possible dimensions (i.e. for n-cube the tree has n leaf).

It has been shown that the access function or routing function to map data on hypercube contains topological properties. This function is convergent in the finite interval. The hypercube is modeled as a discrete space with discrete time because the processor's are in Hamming distances. Some basic properties of hypercube and PDN have been discussed in this paper. We have shown some topological properties of perfect difference network and compared them with the corresponding properties of hypercube. It has been proved that the perfect difference network is not an instance of the generalized hypercube. A scheme of mapping of matrix into a perfect difference network for sparse linear system has been presented here. The complexity of these parallel algorithm is the same as the complexity of the algorithm mapped on a hypercube, which is $O(\log^3 n)$ multiplication / division and $O(\log^2 n)$ for storage and $O(\log^3 n)$ for addition and subtraction. We have used two phases of factorization namely triangular factor and symbolic factor for the efficient solution of sparse linear systems in parallel.

**References:**

1. Ammon, J., "Hypercube Connectivity within cc NUMA architecture" Silicon Graphics, 201LN. Shoreline Blvd. Ms 565, Mountain View, CA 94043.
2. Dekel E.D., Nassimi and S. Sahni, "Parallel matrix and graph algorithms", SIAM Journal on computing, vol 10, No 4, Nov PP 657-675 1981.
3. George Alan and Ng Esmond, "Symbolic Factorization for Sparse Gaussian Elimination with Partial Pivoting" SIAM J. Sci, STAT, COMPUT, Vol8, No. 6, Nov 1987.
4. Katare, R.K., Chaudhari, N.S., "A comparative study of Hypercube and perfect difference network for Parallel and Distributed system and its application to sparse linear system." Varahmihir Journal of computer and information sciences volume 2, Sandipani Academy, Ujjain,(M P ) India, p. 13-30, 2007.
5. Katare, R.K., Chaudhari, N.S., "An attempt to map sparse linear systems on perfect difference network for Parallel and distributed system, Varahmihir Journal of computer and information sciences volume 3, Sandipani Academy, Ujjain,(M P) India, p. 1-10, 2008.
6. Katare, R.K., Chaudhari, N.S., "Study of Parallel Algorithms for sparse linear systems and different interconnection networks".Journal of computer ,mathematical science and Applications, serial publications, New Delhil 2008 (Under Print)

7. Katare, R.K., Chaudhari, N.S., "Some P-RAM Algorithms for sparse linear systems", Journal of Computer Science, USA 2008 (Under Print).
8. Parhami B and Rakov M.A. "Performance Algorithmic and robustness Attributes of perfect difference network" IEEE transactions on parallel and distributed systems, Vol 16, No. 8, August 2005.
9. Parhami B. and Rakov M.A., "Perfect difference networks and related interconnection structures for parallel and distributed systems" IEEE transactions on parallel and distributed systems, Vol. 16, No. 8, August 2005.
10. Quinn M.J., "Parallel Computing", McGraw-Hill INC, 1994.
11. Rakov A Mikhail, "Method of inter connection nodes and a hyperstar interconnection structures" Vs patent 5734280 issued on March 1998.
12. Saad, Y. and Schultz, M.H., "Topological prosperities of Hypercubes" IEEE transactions on computers, Vol. 37 No. 7 pp 867-872, July 1988.
13. Sherman A., "On the efficient solution of sparse systems of linear and nonlinear equations", Ph.D. thesis, Yale University, New Haven, CT, 1975.
14. Tamiko, Teil, "The design of the connection machine" Design Issues, Volume 10, Number 1, spring 1994.
15. Yves Robert, “The impact of Vector and elimination Algorithm", halsted press, a division of John Wiley & sons, New York 1990.